

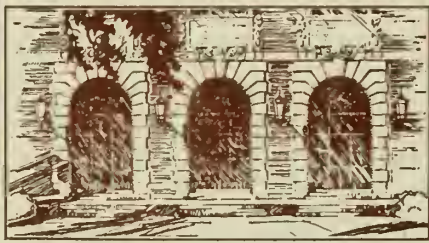
LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84

Ilg6r

no. 824-829

cop. 2



The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

OCT 19 1977

OCT 1 1977

o. 824
20p 2

FROG (FORMULATION OF A ROBOT ORGANIZATION THAT GENERALIZES):

A HARDWARE EXPERIMENT IN MACHINE COGNITION

by

Debasish Bose

August 1976



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

The Library of the
DEC 10 1976
University of Illinois
at Urbana-Champaign



Digitized by the Internet Archive
in 2013

<http://archive.org/details/frogformulationo824bose>

FROG (FORMULATION OF A ROBOT ORGANIZATION THAT GENERALIZES):

A HARDWARE EXPERIMENT IN MACHINE COGNITION

BY

DEBASISH BOSE

August 1976

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

This work was supported in part by the Department of Computer Science at the University of Illinois, Urbana-Champaign, and was submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering at the University of Illinois.

ACKNOWLEDGEMENT

610.84
IL6r
no. 824-827
cap 2

The author wishes to express his gratitude to his advisor, Professor S. R. Ray, for suggesting this thesis topic and for his continued guidance and friendship. He is also very grateful to Professor W. J. Poppelbaum for the opportunity to work in the Information Engineering Laboratory and for his support, advice and understanding.

A special word of thanks is due to Frank Serio for his invaluable suggestions during the design of the mechanical assembly of the project and for his precision construction, in the Electronic Shop, of the various parts. Sam McDowell and Roger Lewis, also of the Electronic Shop, did an outstanding job in the fabrication of the various printed circuits and the panel layout. The author acknowledges their help.

In addition, the author wishes to offer his sincerest thanks to Kathy Gee for the patience and skill with which she typed the thesis, and to Stanley Zundo for his very personal care in drafting the figures.

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
2. CENTRAL PRINCIPLES	6
2.1 Cognitive Mechanisms: Definition of Terms	6
2.2 Learning Strategy	8
2.3 Recognition Strategy	11
2.4 Evaluation of Correlation Values	12
3. SYSTEM DESCRIPTION	15
3.1 The Set-up	15
3.2 The Stimuli	15
3.3 The Decisions	20
3.4 Data Flow Structure	22
3.5 The Front Panel	22
3.6 The Board	26
3.7 The Movable Block and Motion Control	26
4. THE BASIC LOGIC ELEMENTS	30
4.1 The Basic Elements	30
4.2 The Memory Elements	31
5. SYSTEM DESIGN	41
5.1 Structure of the Memory	41
5.2 Memory Integration Unit	45
5.3 Decision Unit	49
6. HARDWARE ORGANIZATION	53
6.1 Machine Representation of Numbers	53
6.2 Hardware Realization of the Basic Logic Elements	54
6.3 Realization of T*-elements	62
6.4 Overall Hardware Organization	70
7. RESULTS AND CONCLUSION	80
7.1 Discussion of a Typical Sequence of Trials	80
7.2 Results in Summary and Concluding Remarks	84
APPENDICES	89
A. Summary of a Sequence of Trials	89
B. Memory Requirement and Response Time for the Proposed Scheme.	94
REFERENCES	96
VITA	97

1. INTRODUCTION

Artificial Intelligence: (a) the ability of machines to organize information into meaningful patterns: ability to recognize, store, recall and manipulate meaningful patterns in playing games, solving problems, answering questions, etc. and in controlling the actions of other mechanisms: (b) the ability of a machine to adapt to its environment, particularly to respond to patterns of stimulations not explicitly foreseen in its design; (c) the observed performance of such machines as measured by comparison with, or in competition against human intelligence.[1]

This attempt to define the objectives of the area of investigation termed Artificial Intelligence (AI) shows that it is quite enormous in its scope. The investigations in this area, to date, seem to have only scratched the surface of a potentially very rich field of research.

On one hand, AI has drawn its ideas from a variety of more or less formal disciplines like logic, information theory, statistical decision theory, behavioral psychology, etc. On the other hand, AI has accumulated within its folds simple ad hoc informal algorithms to solve specific problems. Workers in AI have tried not to be prejudiced toward making their systems strictly biological or humanoid. That is the reason why AI has developed into a discipline by itself. But it must be added that "although AI is not aimed at replicating the organization of a biological organism but only its behavior, yet any insight provided by psychological introspection should be profitably used." [2]

A cognitive¹ mechanism as an artificial intelligence scheme may be described as follows (see Figure 1.1). A stimulus of some kind (a scene, a symbol, a sentence, etc.) is first preprocessed by a sensor which performs

¹In the sense of psychologists; meaning, a response judgement requiring an extended time "consideration" of the stimulus data as opposed to a recognition task in which the response is relatively straight-forward and automatic.

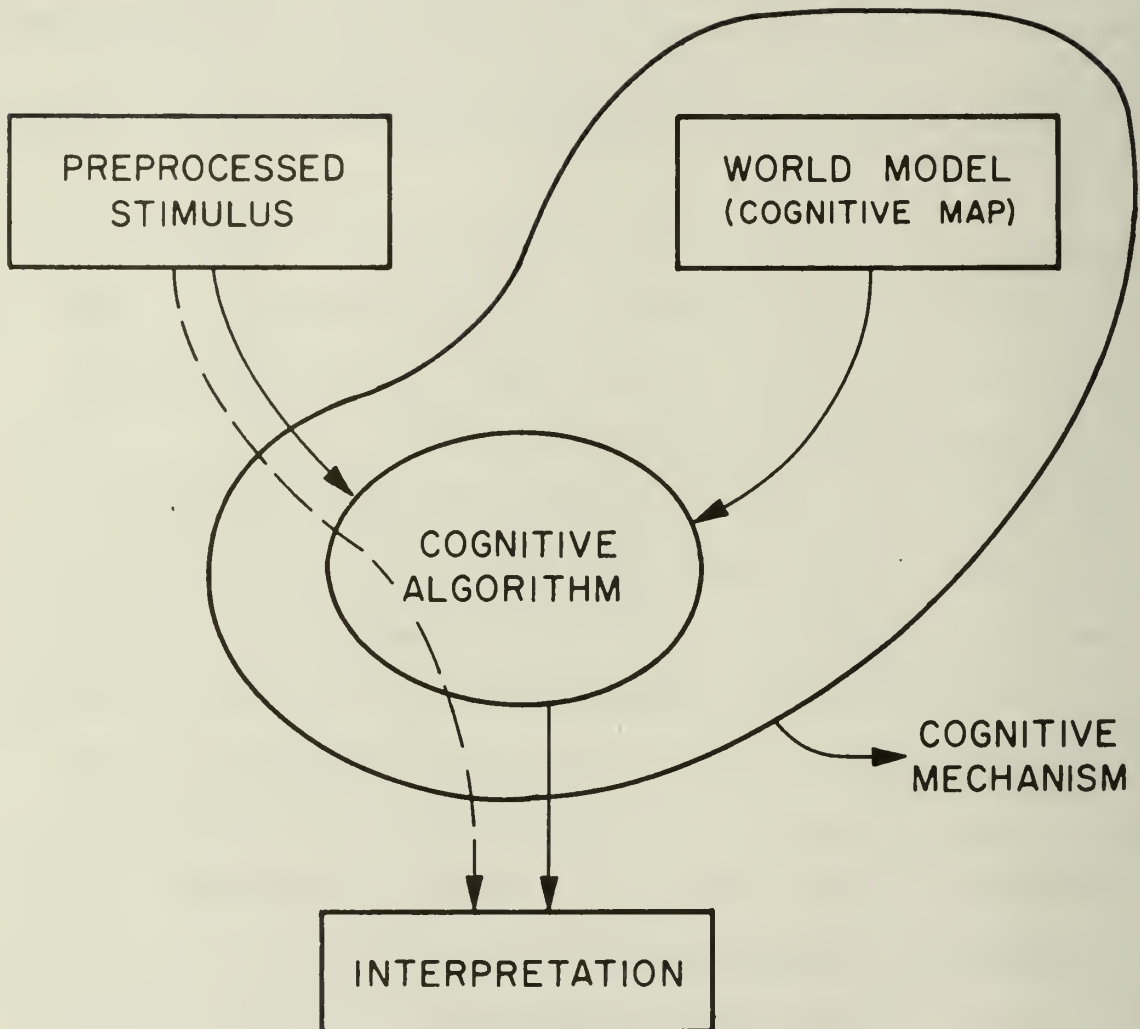


Figure 1.1 General Cognitive Mechanism

an unbiased transduction (like that performed by a TV camera, or a microphone). The cognitive mechanism performs a more subtle, and not unbiased, kind of transduction to produce an interpretation. The cognitive mechanism contains two basic constituents: a model of the world and an algorithm (activity) which performs the interpretation of the stimulus under the guidance of the model. The conjoint of the map and the algorithm may appropriately be called a "cognitive memory" in that it exhibits both the static (passive) property of storing data and the dynamic (active) property of recognizing stimuli. The product of the cognitive activity is, as we have already mentioned, the "interpretation" which can be characterized as the linking of features of the stimuli to features of the map.

The cognitive map may be looked upon as a set of competing hypotheses or concepts (like the descriptions of abstract objects and relations in the case of systems analyzing natural scenes) in a complex relational hierarchy or heterarchy and constitutes what can be called a "world model". In most cases, this world model is adaptive in nature where the model learns from and is under constant revision based on the experience of the system with known inputs.

Often, the central problem in an AI system is to develop a "world model" based on experience, so that an interaction of the perceived stimulus with the model result in some sort of identification. Also important is the incorporation of a mechanism to revise the world model continuously in order for it to be better adapted to a better understood situation. Talking about worlds models (or "micro-worlds" as they choose to call it) Minsky and Papert assert:

One cannot overemphasize the importance of this kind of scenario both in technical and in everyday thinking. We are absolutely dependent on having simple but highly-developed models of many phenomena. We see this problem--of using schematic heuristic knowledge--as a central problem in Artificial Intelligence.[3]

The existence of a world model is supported, for example, by the classical research of Jean Piaget [4]. The concept has since been postulated in various forms by several investigators in the area of psychology and artificial intelligence (see, e.g. [5], Ch. 4). More recent evidence has been presented by Eric Lenneberg [6, Ch. 7-9]. He concludes that the basis of human communication is a model of the constraints and relationships among perceived elements in the sensory environment. His conclusions indicate that the world model processes a sensory input from the general (i.e., gross context) to the specific (i.e., toward more minute detail) by stimulating successive states corresponding to inferred conditions and relationships.

Phenomenologists and Gestalt psychologists have observed that recognition of ordinary spatial and temporal objects by living beings seem to proceed from indeterminate, global anticipation to more specific and detailed search. In human speech recognition, for example, it is observed that the phoneme comes after the fact; it is constructed, if at all, as a consequence of perception not as a step in the process of speech recognition. In his analysis of speech recognition Oettinger, therefore, concludes:

This drives me to the unpopular and possibly unfruitful notion that maybe there is some kind of Gestalt perception going on, that here you are listening to me, and somehow the meaning of what I'm saying comes through to you all of a piece. And it is only aposteriori, and if you really give a damn, that you stop and say, "Now, here was a sentence and the words in it were of such and such type, and maybe here was a noun and here was a vowel and that vowel was this phoneme and the sentence is declarative, etc." [7]

Pattern recognition schemes, based on searching through lists of isolated traits, although works moderately efficiently when applied to small models, fail to work as efficiently when applied to large ones, mainly because of the enormous amount of time spent in searching through large volumes of data. It appears that the world is experienced as fields within fields. Bits and aspects of objects in the world are not experienced as isolated facts, but as nested series of contexts.

This paper develops a formalism for the adaptive development of a world model in an uncertain environment suitable for efficient cognition in accordance with the principle of "processing from the general to the specific."

The model is then employed to simulate the feeding habits of a small animal where the device sustains existence by trying to recognize objects based solely on visual clues in an artificial world with sources of potential food supply and hazards. This hardware demonstration system is called FROG (Formulation of a Robot Organization that Generalizes).

2. CENTRAL PRINCIPLES

2.1 Cognitive Mechanisms: Definition of Terms

Consider a cognitive mechanism (of the type mentioned in Chapter 1) operating in an environment consisting of various classes of stimuli (objects to be recognized). Each class of stimulus is uniquely characterized by the value of its feature or sensory input vector $S_1 S_2 \dots S_n$, where n is an arbitrary dimension. In order to allow for variations within the same class a small amount of spread is allowed around each value. The sensory inputs $S_1, S_2 \dots S_n$ are derived from one or more sensors on the mechanism. A certain amount of preprocessing is assumed in the generation of the sensory inputs. Example of sensory inputs are parameters like color, relative shape, relative size, distance, position etc. These inputs are allowed to take on values in the signal continuum 0 to 1, i.e., they are fuzzy sets. The process of assigning a numerical value (or a membership grade, as it is called in the theory of fuzzy sets [8]) to an actual feature value is likely to be subjective.

Assume that a learning encounter of the mechanism with a stimulus (e.g., an animal trying to feed on an object) is associated with a multi-parameter reward vector. The reward vector may be made up of parameters (fuzzy sets) such as taste value, pain inflicted etc. Each class of stimulus again has a unique reward vector.

Figure 2.1 shows a schematic diagram (more detailed than that in Figure 1.1) of such a cognitive mechanism. Every presentation of a stimulus to the mechanism first induces a recognition (or read) cycle where some response values are generated by a cognitive algorithm acting on the sensory inputs. The response values are anticipated reward-parameter values or functions

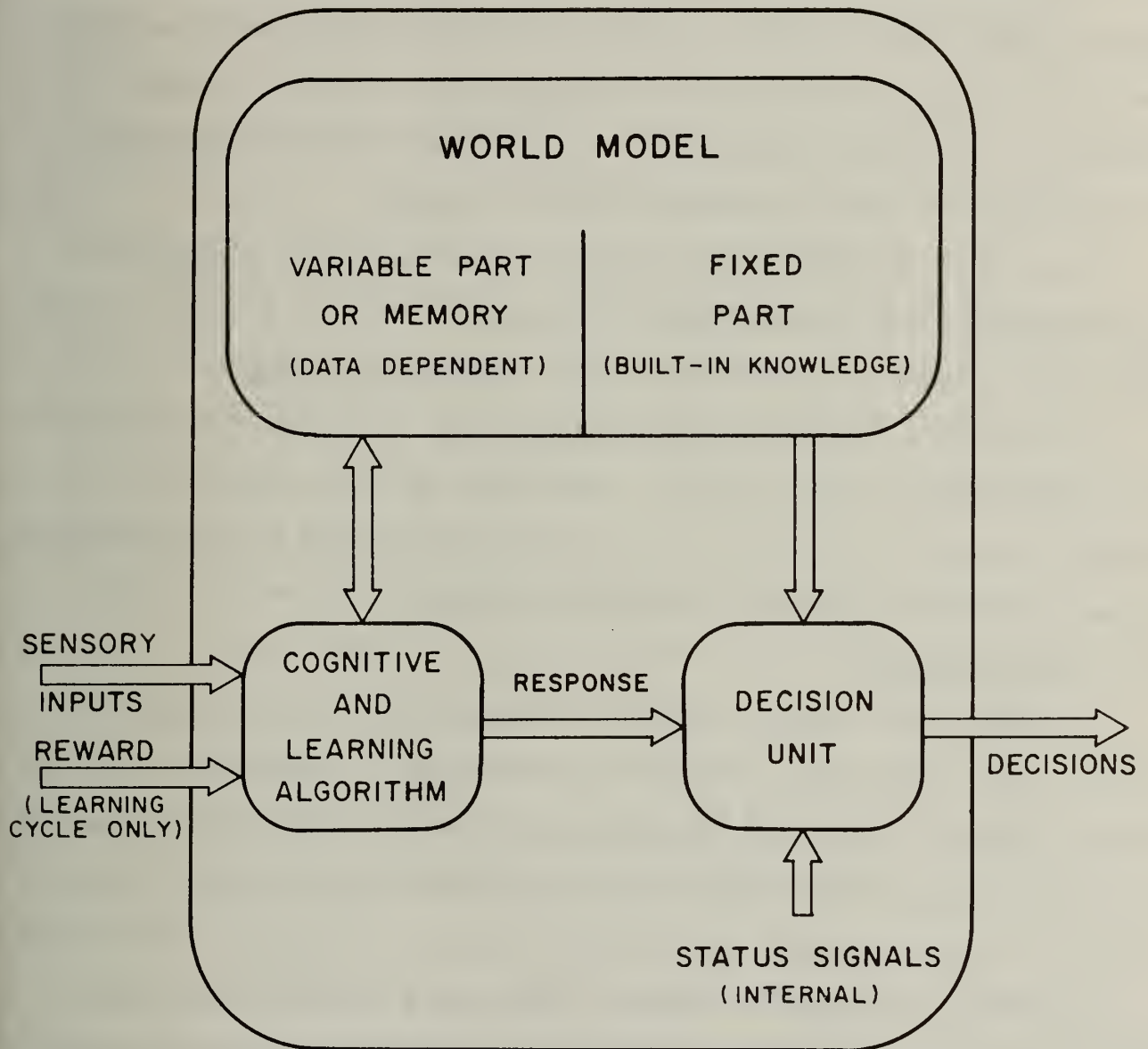


Figure 2.1 Cognitive Mechanism: A Modified Schematic Diagram

thereof. The generation of the response values is assisted by the variable part of the world model or memory, which accumulates the experience with elements of the sensory world. These response values together, possibly, with other status signals (such as, somatic signals like tiredness, hunger etc.) are processed by a decision making algorithm or decision unit to arrive at some decision(s). The decision making is, of course, influenced by the fixed part of the world model which incorporates built-in reflexes.

Depending on the decision(s), there may or may not follow a learning (or write) cycle when the reward value is inputted to the device for an augmentation of the world model according to a learning algorithm.

In a generalized situation each individual part in Figure 2.1 may be postulated to affect every other part. The domain and the direction of influence implied by the arrows in the diagram are sufficient for the particular situation simulated in FROG as a cognitive mechanism.

2.2 Learning Strategy

To explain the learning strategy let us assume the context of a small animal trying to avoid the bad tasting stimuli amongst its cohabitants. The various classes of stimuli are uniquely specified by the three sensory inputs S_1 , S_2 and S_3 (e.g., three visual features like color, shape and size). The reward is a single parameter; the "bad taste" value of a stimulus.

At birth, the device will attempt to feed upon a relatively broad range of objects. Each feeding attempt is followed by internally generated signals measuring the tastiness of the food object. Upon encountering a very bad tasting object O_1 with inputs S_1^* , S_2^* and S_3^* the world model adjusts strongly to reject as food any further objects having stimulus S_1^* and S_2^* and S_3^* , to reject less strongly any object having double joint combinations of the stimulus inputs, to reject even less strongly the input values occurring

individually. If some other object, O_2 , has a bad taste and shares some properties with O_1 (say the joint occurrence of S_1^* and S_3^*), then its effect is to strengthen the rejection of $S_1^* \cdot S_3^*$ as a property associated with good food, thereby tending to generalize that this joint property implies "inedibility."

Figure 2.2 illustrates the basic concept in a three-dimensional space. The darkness of shading is proportional to the strength of rejection of stimulus $[S_1^*, S_2^*, S_3^*]$ due to single occurrence associated with a negative result (e.g., a bad taste). The occurrence of a negative result associated with the inputs $[S_1^*, S_2^*, S_3^*]$, for example, will add increased rejection to the pair $S_1^* \cdot S_3^*$ as well as to the S_1^* and S_3^* volumes. In other words, evidence that the stimulus $S_1^* \cdot S_3^*$ is associated generally with "bad taste" is being accumulated.

This multi-dimensional correlation space (matrix) may be thought of as a filter. Sensory inputs processed through this filter results in the response. This filter, therefore, constitutes the variable part of the world model. The adaptive development of the filter is made faster for multiple joint occurrences with the result that the device tends to reject large fractions of the stimulus range as food only when evidence has become quite strong.

It is to be noted that, in a real-life pattern recognition situation, the feature space will typically have more dimensions than three. In this discussion, as well as in the illustrative simulation, FROG, the problem is restricted to a 3-dimensional situation merely to restrict the complexity of the learning process without sacrificing or simplifying the inherent difficulties of the problem. The approach is perfectly general and can be extended to any dimension.

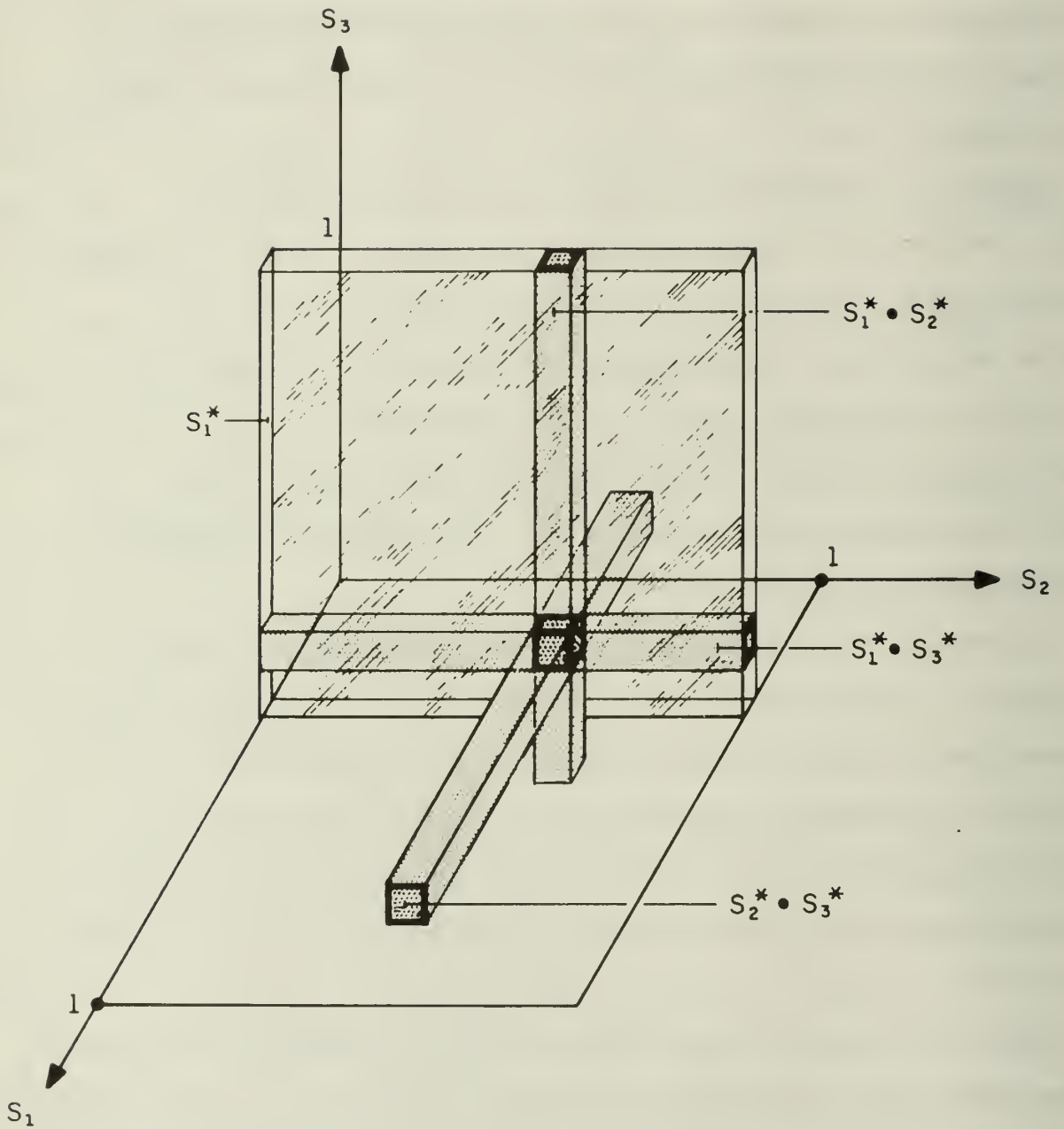


Figure 2.2 Adaptive Filter:
Relation to a Stimulus Space of 3 Dimensions

If the reward is multi-parameter, a separate correlation space has to be developed for each parameter of the reward.

2.3 Recognition Strategy

As mentioned before, the cognitive activity in the biological world seems, largely, to proceed from the general to the specific. The order of processing seems to be intimately connected to the fact, that in order to survive, living organisms have to react quickly. The recognition strategy described below is in accordance with this observation.

When a stimulus is presented to the mechanism, it tries, first, to produce a response by examining the sensory inputs taken individually. It refers to its world model to see what it knows about each of these sensory input values taken individually, without paying attention to the particular combination in which it appears. The strongest correlation values corresponding to the sensory inputs determine the response. If this response is not adequate (according to some set criteria) for a decision, the memory is consulted for response based on the sensory inputs taken in pairs. Further, if necessary, the data base in the world model is looked up for what is known about the inputs taken three at a time and so on.

Obviously, the number of possibilities at the first (the most general) level is much smaller than that at the second level, and very much smaller than that at the third level and so on. For example, consider the case where there are only three sensory inputs S_1 , S_2 and S_3 , and a single parameter reward R . If for each of these signals ten different values are recognized, then there are a maximum of 300 correlation values (10×10 along each axis) to remember at the first level, a possible 3000 entries in the second level and a list of as many as 10000 values to search through in the third or the most specific level.

This strategy makes the search more efficient at the risk of making errors due to over-generalization every once in a while. Also, the system is capable of handling new situations with partial matches.

2.4 Evaluation of Correlation Values

This section specifies the quantitative evaluation of the correlation values between various unique combinations of the sensory input values and the corresponding values of the reward input.

The memory (see Figure 2.1) is hierarchial in nature with various levels. In the top level (level 1) the sensory inputs are associated individually with the reward values. Level 2 retains experience corresponding to the double joint combinations of the sensory inputs and so on.

As mentioned before, a separate stimulus-response correlation space (to be thought of as a 'submemory') has to be developed for each parameter of the reward. We, of course, will concern ourself with the elements of only one submemory with sensory inputs $S_1, S_2 \dots S_\ell$, and one reward input R .

As the cognitive mechanism has more and more learning encounters with various stimuli, let us assume, that at any point in time, corresponding to a particular value S_i^* of the sensory input S_i ($i=1,2,\dots$ or ℓ), it comes across the reward value R^* a total of m^* times. This experience can be summarized by storing in an unit memory cell in the most general level (level 1) a numerical correlation value of the form:

$$\{\min(1, m^*/n_1)\} \cdot \{f_1(S_i^*) \cdot R^*\}$$

when n_1 is the number of encounters necessary for full learning in level 1 (further explanation follows). The function f_1 should be chosen appropriately in order for the sensory input value S_i^* to have a desired influence on the correlation value. Suggested forms of f_1 are: $f_1(S_i^*) = \text{constant} = 1$ (say) or $f_1(S_i^*) = S_i^*$.

The rationale behind this choice for the form of the correlation value is as follows. The use of the product as a correlation function implies that more striking the feature (i.e., larger the sensory input value) and/or higher the reward value, stronger is the experience.

The factor $\min(1, m^*/n_1)$ is a confidence function which makes it necessary to have n_1 repetitions (write cycles involving the same input values) for the correlation value to achieve full potential.

So, a level 1 memory cell stores the following quantities: A value each of S_1 , R and m , plus the correlation value. n_1 is a built-in parameter determined by the location (i.e., the level) of the cell within the memory.

During any recognition (or read) cycle, if the mechanism comes across a stimulus with the value S_1^* for the sensory input S_1 , then this memory cell in level 1 outputs the current correlation value.

Similarly, in the second level (level 2), a unique combination S_p^* , S_q^* of values of two sensory inputs S_p and S_q ($p, q = 1, 2, \dots, p \neq q$) and the corresponding value R^* of the reward, sets a level 2 memory cell whose output for the same values S_p^* and S_q^* input to the cell during a read cycle is:

$$\{\min(1, m^*/n_2)\} \cdot \{f_2(S_p^*, S_q^*) \cdot R^*\}$$

where m^* is the number of encounters with stimuli having a value R^* for the reward and the values S_p^* , S_q^* for the pth and the qth sensory inputs, and n_2 is the number of encounters needed for the correlation value to reach full potential. The function f_2 can be specified to be a convenient function of the sensory input values. In FROG, the nonlinear function of 'maximum' is used for f_2 .

A memory cell in level 3 sets to a new set of three input values S_x^* , S_y^* and S_z^* ($x, y, z = 1, 2 \dots l$, $x \neq y \neq z$) and R^* to form the correlation value:

$$\{\min(1, m^*/n_3)\} \cdot \{f_3(S_x^*, S_y^*, S_z^*) \cdot R^*\},$$

the quantities m^* and n_3 being defined in a manner similar to the above cases.

In FROG, $f_3(S_x^*, S_y^*, S_z^*) = \max(S_x^*, S_y^*, S_z^*)$.

For repetitions and read-outs, it is not necessary to have an exact match between the presented values and the stored values. Some spread ($+\Delta R$ in the sensory input values, $+\Delta R$ in the reward values) may be allowed around the stored values to satisfy the condition of a match.

Note, that by specifying $n_1 > n_2 > n_3$ etc. we can assure that multiple joint occurrences are adapted more quickly than single occurrences of the sensory inputs. Also, note, that the memory cells in the different levels have different numbers of sensory inputs.

The details of the structure of the memory and how the outputs of the memory cells, during a recognition cycle, can be used to provide the response will be illustrated in discussing the system FROG later.

3. SYSTEM DESCRIPTION

3.1 The Set-up

FROG is an autonomous mini-robot, whose objective is to sustain its existence. It is a movable block which is capable of moving back and forth between several fixed locations on a demonstration board under the on-line control of a special-purpose computer to which it is connected by a cable (Figure 3.1). Whenever it is alarmed, FROG takes refuge in one of the locations, which is its safe resting place. At other locations any one of a fixed number of prey (food with varying taste) and enemies is presented to FROG in a specified manner. In order to feed upon the desirable stimuli and to avoid the not so desirable or the dangerous ones, FROG has to move between the locations.

FROG is used to simulate the nature of errors and the behavioral traits shown by the proposed learning scheme during the learning phase (remember that some repetition of encounters is necessary for full learning). The behavioral pattern will, of course, be a function of the sequence in which the stimuli are encountered and the extent to which the stimuli shares features amongst each other.

3.2 The Stimuli

The cohabitants of FROG's environment are some prey (edible and inedible) and enemies. The machine learns to discriminate between these by means of their visual characteristics. According to Maturana et al. in the "Anatomy and Physiology of Vision in Frog"[9] the ganglion cells in the retina of a frog form five natural classes which break down every image into five main characteristics which are transmitted to the brain.

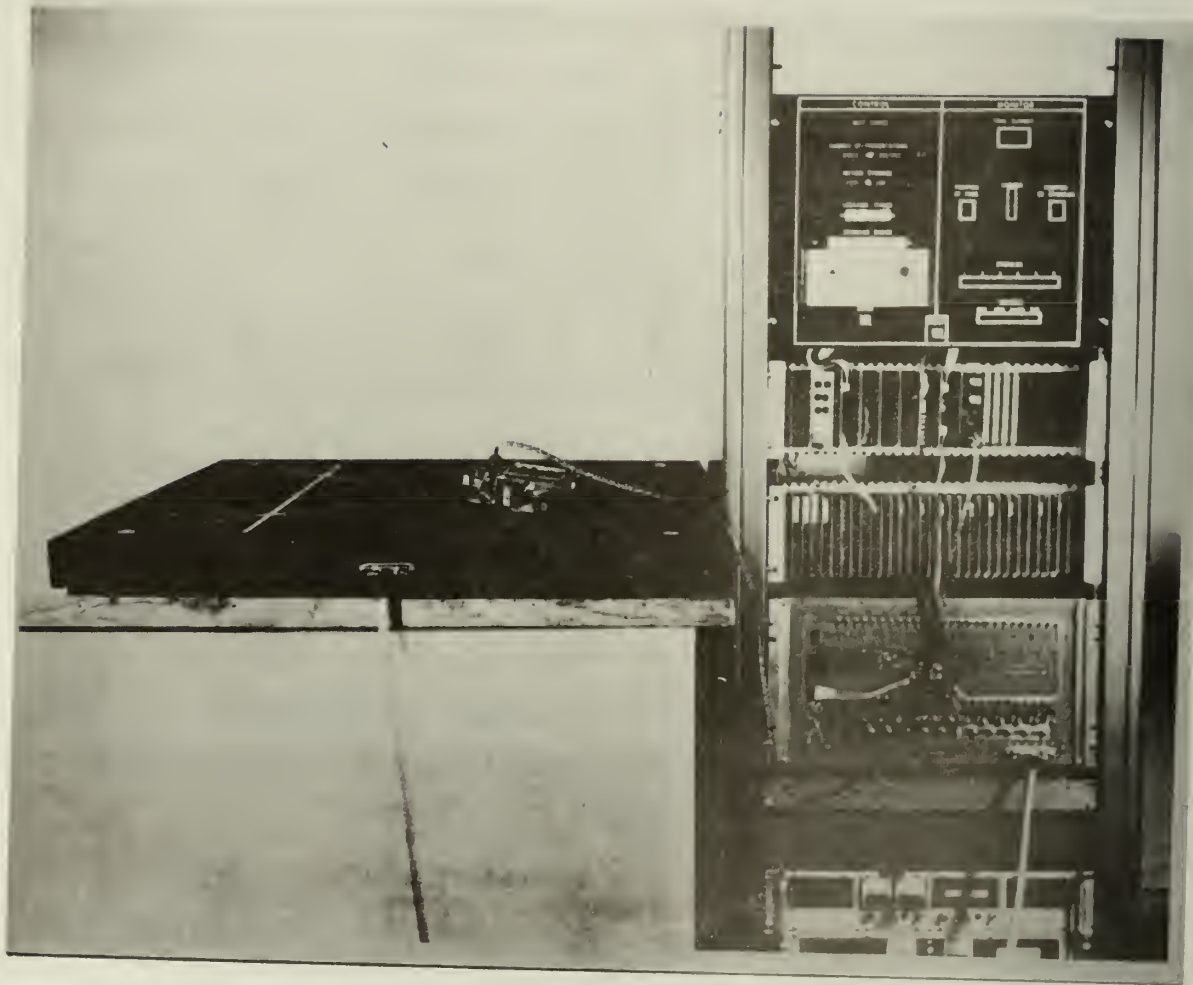


Figure 3.1 The Set-up

Four of them act on the visual image to perform complex analytical operations that remain invariant under changes of the general illumination and changes of the general outlook of visible environment, the fifth class measures light intensity. These operations of the ganglion cells are briefly described by their names:

- Class 1 Sustained edge detection--presence of sharp edges.
- Class 2 Convex edge detection--presence of convex dark edges.
- Class 3 Changing contrast detection--presence only of changing patterns.
- Class 4 Dimming detection--occurrence of adequate darkening of the receptive field.
- Class 5 Darkness detection.

The brain of a frog utilizes these sensory signals along with other predicates for its cognition of the objects around.

Without a loss of generality, we assume that in the artificial world of FROG the various classes stimuli are uniquely specified by the following visual characteristics:

- i) color,
- ii) size (relative),
- iii) shape (relative).

It is assumed that the total spectrum for each of these attributes is mapped onto the interval 0 to 1 by the eye of FROG. So for each stimulus presented to FROG, these numerical values (each in the range 0 to 1) are input to FROG specifying the color, the relative size and the relative shape of the stimulus. Internally FROG generates a signal which is the measure of its hunger.

There are three distinct types of stimuli. The first is the class of good tasting ones which are associated with a reward parameter called the good taste value, $g.t.$ ($0 \leq g.t. \leq 1$). The second is the class of bad tasting stimuli for which the reward is specified as a bad taste value, and the third is the class of enemies which inflicts pain to a varying degree as the reward (in a negative sense, of course).

Two stimuli of each type are chosen. Figure 3.2 shows the symbolic representation of the various types of stimuli. The sensory (visual) input and reward values assigned to the various stimuli are based on what the author feels are reasonable and realistic choices. The top half of Table 3.1 lists the assigned values which are to be interpreted as 4-bit binary fractions. The types of stimuli which share the same input values at various levels with the listed type, are shown in the lower half of the Table for each of the six types. It is apparent that the given choice provides enough basis for confusion in the general level (level 1). There are some confusing similarities in the second level. In the third level since all the details are considered at a time, there is no confusion.

3.3 The Decisions

The central mechanism of FROG for adjusting to the idiosyncrasies of its environment is the world model. A "situation analysis" is formed by the "decision unit" which combines the visual data, interpreted through the world model, and hunger signal, which measures the requirement of FROG for food, and decide on a course of action. The four modes of activity or decisions are: 1) feed; 2) rest; 3) alarm; 4) flee (see Figure 3.3).

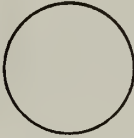
	GOOD TASTING		BAD TASTING		DANGEROUS	
	1	2	3	4	5	6
SYMBOL (APPROXIMATE VISUAL APPEARANCE)						
COLOR	BLACK	YELLOW	RED	YELLOW	BLACK	RED
SHAPE	TRIANGULAR	LONG & THIN	ROUND	TRIANGULAR	ROUND	LONG & THIN
SIZE	MEDIUM	LARGE	SMALL	MEDIUM	LARGE	LARGE

Figure 3.2 The Stimuli: Symbolic Representation

Table 3.1 Assignment of Values to the Feature Inputs of the Stimuli

	STIMULUS TYPE					
	1	2	3	4	5	6
NONVISUAL CHARACTERISTICS						
GOOD TASTE	1111	1110	0000	0000	0000	0000
BAD TASTE	0000	0000	1111	1110	0000	0000
PAIN	0000	0000	0000	0000	1110	1111
VISUAL CHARACTERISTICS						
COLOR	1101	1011	1111	1011	1101	1111
SHAPE	1100	1110	1011	1100	1011	1110
SIZE	1100	1110	1011	1100	1110	1110
LEVEL 1						
S ₁	5	4	6	2	1	3
S ₂	4	6	5	1	3	2
S ₃	4	5,6		1	2,6	2,5
LEVEL 2						
S ₁ , S ₂						
S ₁ , S ₃						
S ₂ , S ₃	4	6		1		1
LEVEL 3						
S ₁ , S ₂ , S ₃						

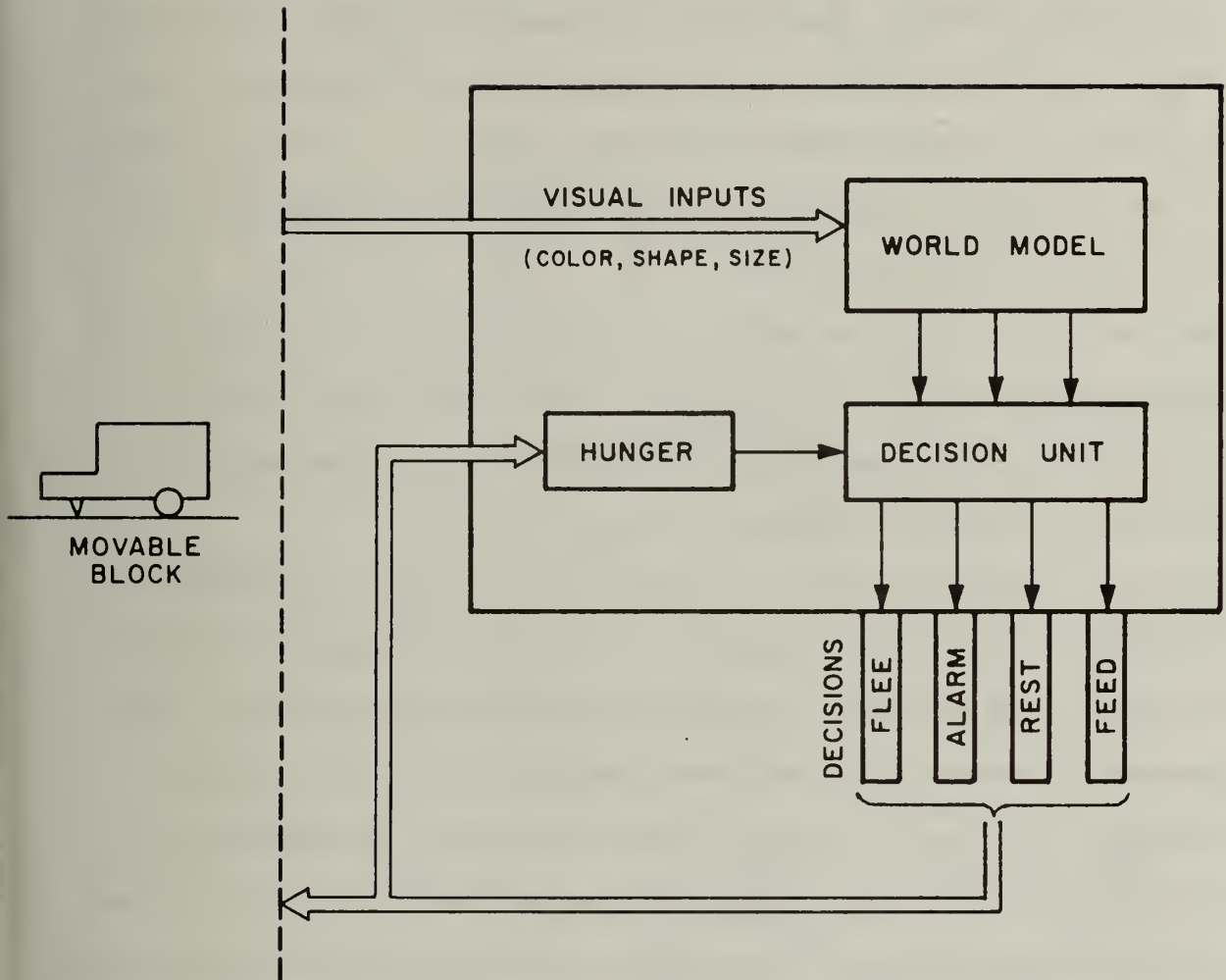


Figure 3.3 Functional Sketch of FROG

3.4 Data Flow Structure

When the sensory (visual) inputs are presented to the MEMORY, its outputs from the various memory elements are processed by the MEMORY INTEGRATION UNIT (MIU) to form the response, the data relevant for decision making. The response value(s), the hunger condition and the information about the position of the stimulus (location signal) interpreted through the DECISION UNIT (DU) result in a decision (see Figure 3.4).

When comparing this diagram with the structure detailed in Figure 2.1 one should recognize 1) the MIU as the recognition algorithm, 2) the DU as the decision algorithm, and 3) the conjoint of the MEMORY and the built-in conditions of DU as the world model.

Figure 3.4 shows the data flow structure of the system. The STIMULUS PRESENTATION UNIT (SPU) can be programmed or manually controlled to decide on the stimulus to be presented. The BOARD is the environment of FROG, where the appearance of a stimulus at one a few fixed locations is indicated by proper displays. The sensory (visual) signals are sent to the MEMORY for a read (note the read/write signal to indicate recognition/learning cycle). The reward values (the good taste, the bad taste and the pain values) are inhibited during read cycles. The location signal goes to the DU. The decision is made known to the MOTION CONTROL UNIT (MCU) and the SPU. If the decision is to feed, the SPU sends the sensory inputs and the reward values for a write into the MEMORY. Miscellaneous status signals go from the SPU to the MCU. MCU controls the movement of the MOVABLE BLOCK (MB) appropriately. The value of the hunger signal is augmented in accordance with the decision.

3.5 The Front Panel

FROG is required to act upon one stimulus at a time, i.e., one stimulus per trial or presentation. The stimulus for a trial and the location at which

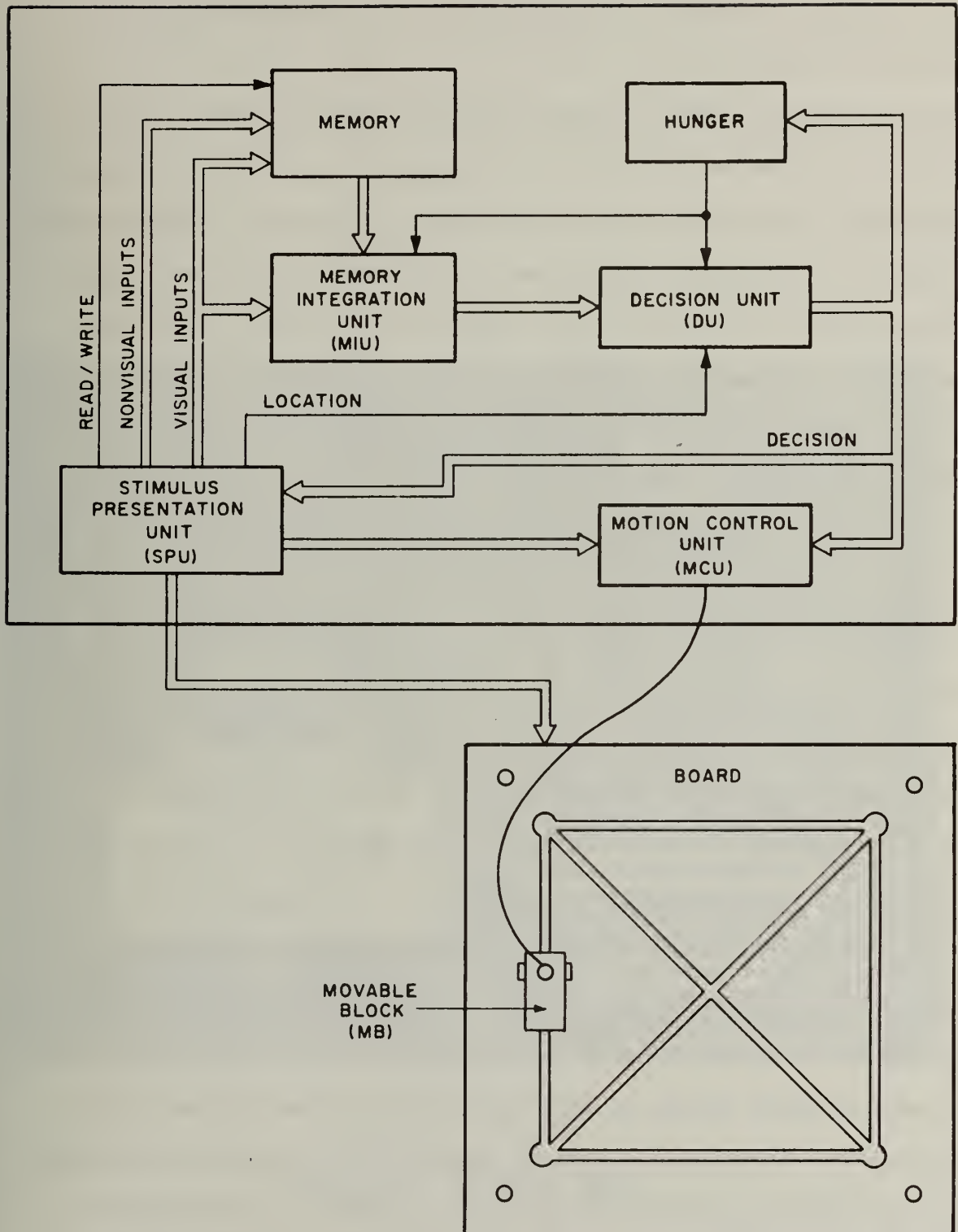


Figure 3.4 Data Flow Structure

it should appear is determined by the switches on the control part of the front panel (see Figure 3.5).

The SPU can be operated in either SINGLE or MULTIPLE mode. In the SINGLE mode, the observer lets his/her choice known to the SPU by pressing the appropriate buttons under the labels STIMULUS CHOICE and LOCATION CHOICE on the panel. On pressing the ENTER switch the cycle of operation begins whereby the visual data is sent to the MEMORY for a read-out. Appropriate displays on the monitor side of the panel indicates the TRIAL NUMBER, the POSITION OF FROG, the LOCATION OF THE STIMULUS, the HUNGER value, the STIMULUS being presented and the DECISION. Indicators also light up on the BOARD to indicate the LOCATION of the stimulus. Depending on the decision, MB may or may not move. The end of the cycle is indicated on the panel and the observer is requested the NEXT CHOICE.

Each cycle takes a lot of time if MB has to move around. In order to show the effect of a large number of trials, it is helpful to be able to disable MB and quickly run through a number of trials. This can be done by throwing the NUMBER OF PRESENTATIONS switch to MULTIPLE. In this mode MB does not move. We can specify an augmentation of the memory by an arbitrary number of trials by dialing in the desired number in the two-digit thumbwheel switch. The observer can specify any one stimulus (1,2,3 or 4) or a random (R) selection of those. The latter can be done by pressing the R button under STIMULUS CHOICE, in which case all the stimuli appear in a random sequence. A random selection of the LOCATION is possible by pressing the R button under the LOCATION CHOICE. The R buttons are not effective in the SINGLE mode. (A random selection in the SINGLE mode can, however, be achieved by going to the MULTIPLE mode and setting the thumbwheel switch to one.) On pressing the ENTER switch, the system automatically goes through all

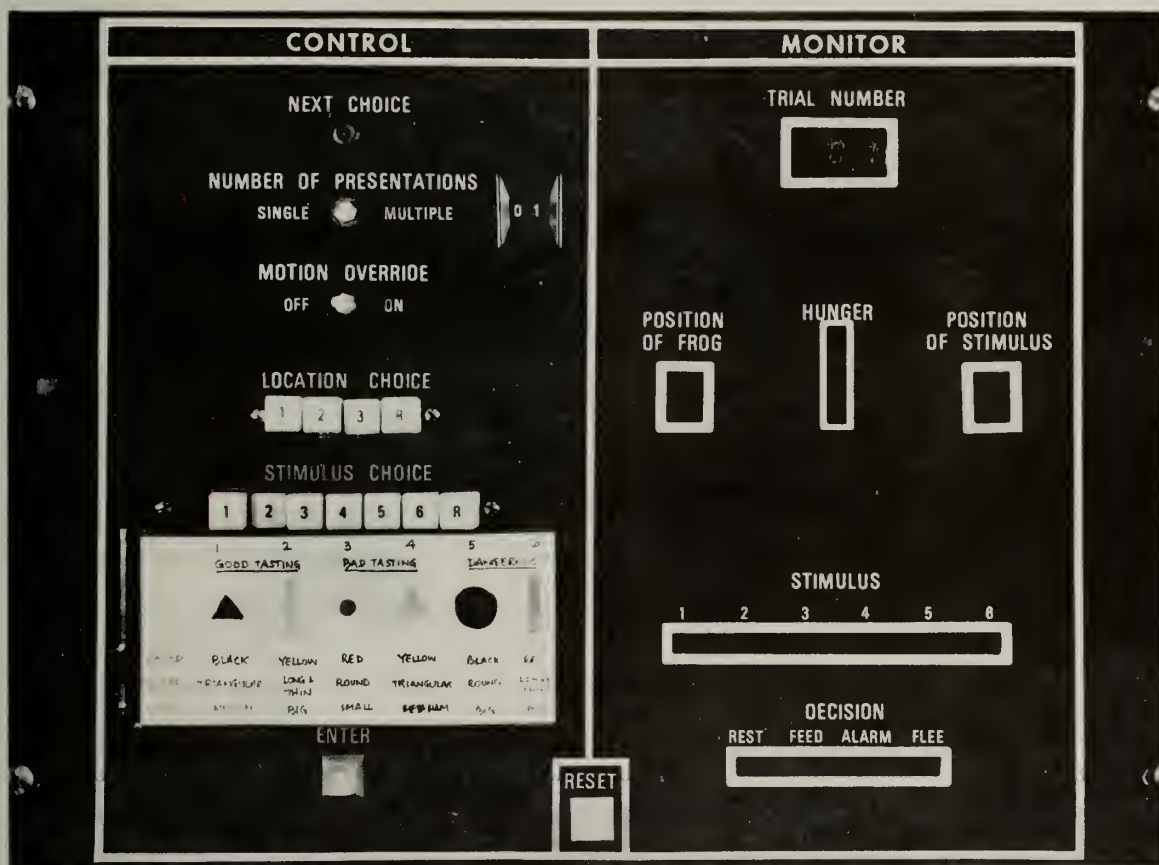


Figure 3.5 The Front Panel

the read and the necessary write cycles, update all internal variables and augment the MEMORY. MB is not moved so that the current position of MB remains fixed. After the requisite number of trials, the NEXT CHOICE indicator comes on.

3.6 The Board

The BOARD (see Figure 3.6) has four locations. Location # 0 is FROG's safe resting place or home. The stimuli can appear in the other three locations. Appearance of a stimulus at any location is indicated by a light lighting up at that location.

The surface of the BOARD is painted black to minimize the light reflected from it. The paths joining the four locations can be lit up individually by light bulbs embedded deep under the track. The usefulness of all this will become evident in the following section when the motion control of MB is discussed.

3.7 The Movable Block and Motion Control

The movable block (MB) is shown in Figure 3.7. It has two independently driven rear wheels and a small caster wheel in the front. This allows MB to be driven forward or backward and also to be turned left or right about an axis within itself. The drive circuits for the motors are on MB. The method by which movement of MB between the locations is controlled is chosen for its simplicity. It is also inconspicuous enough that the control method is not immediately apparent, visibly or audibly, to a casual observer. This requirement could have been better satisfied by other methods (like radio control), but that would have added unnecessary complication to the system.

The motion of MB between the two locations is controlled by lighting up appropriate segments of the track from underneath and using a control system employing photocells at the front end of MB. The control is programmed to turn MB in a counterclockwise direction to hunt for the illuminated

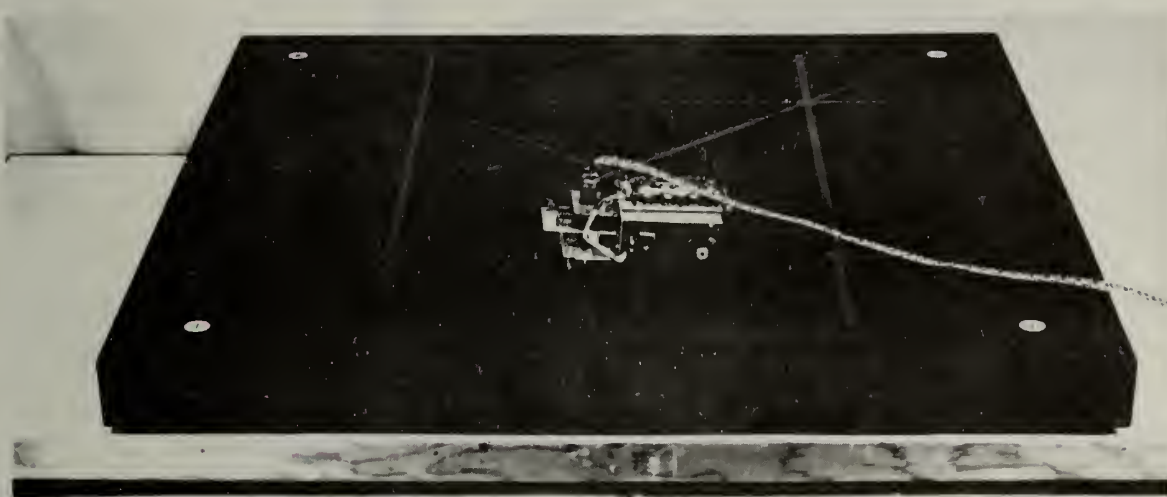


Figure 3.6 The Board

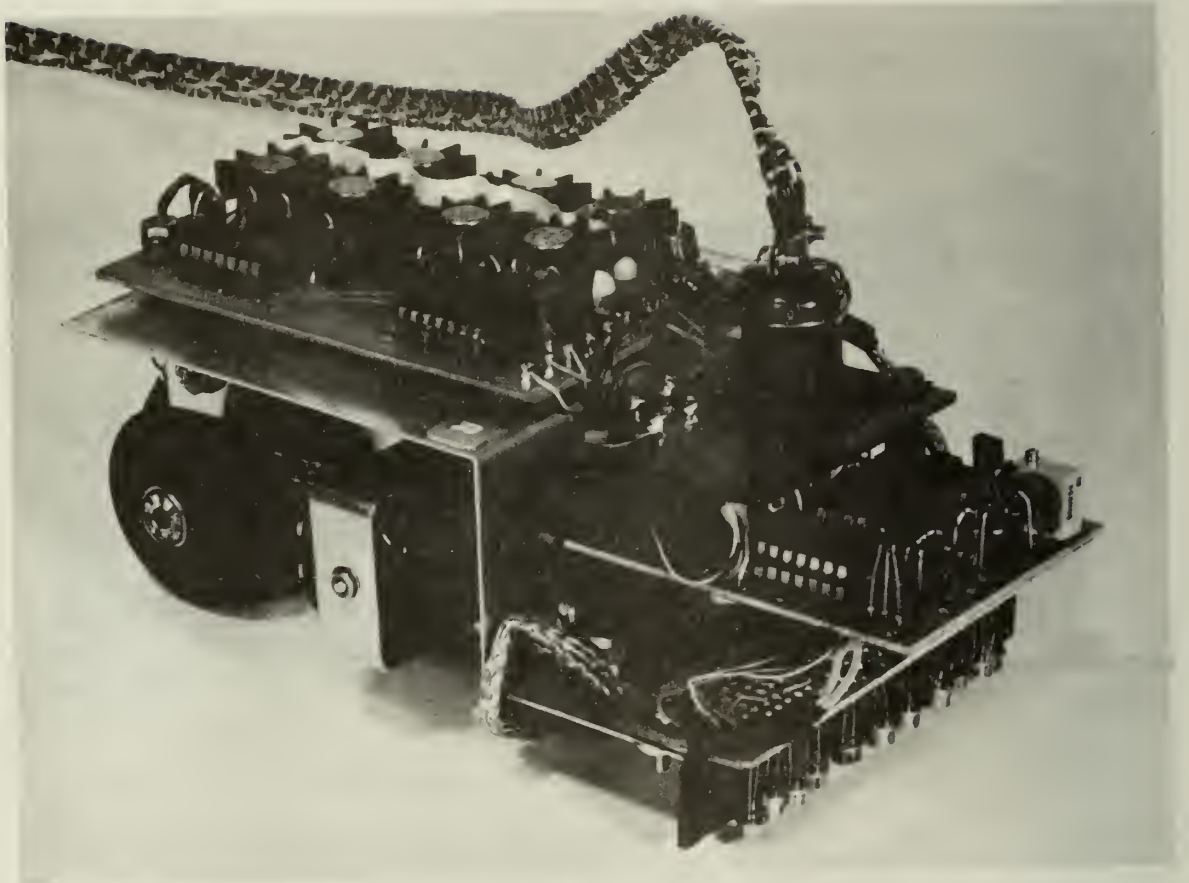


Figure 3.7 The Moveable Block

track. Once the photocells are over the lighted track, the control makes MB follow the track. The intensity of the light is higher at the end of the track which is detected to signal the stopping of MB on reaching the destination location.

4. FROG: THE BASIC LOGIC ELEMENTS

4.1 The Basic Elements

All of the logic elements in FROG are drawn from a set of nine basic elements, defined hereafter, which operate in the signal continuum 0 to 1. All inputs, x_i , and outputs, y , have values in the continuum unless noted otherwise.

<u>Element Name</u>	<u>Symbol</u>	<u>Nearest Analog</u>	<u>Definition</u>
Complement (C)	$x \rightarrow \textcircled{C} \rightarrow y$	NOT	$y = 1 - x$
Minimum (m)	$\begin{matrix} x_1 \\ \vdots \\ x_k \end{matrix} \rightarrow \textcircled{m} \rightarrow y$	AND	$y = \min_{i=1}^k (x_i)$
Maximum (M)	$\begin{matrix} x_1 \\ \vdots \\ x_k \end{matrix} \rightarrow \textcircled{P} \rightarrow y$	OR	$y = \max_{i=1}^k (x_i)$
Product (P)	$\begin{matrix} x_1 \\ x_2 \end{matrix} \rightarrow \textcircled{P} \rightarrow y$	--	$y = (x_1) \cdot (x_2)$
Equivalence (E)	$\begin{matrix} x_1 \\ x_2 \end{matrix} \rightarrow \textcircled{E} \rightarrow y$	Equiv. (fuzzy)	$y = x_1 \text{ if } x_1 - x_2 \leq 0.05$ $y = 0 \text{ otherwise}$
Selective	$\begin{matrix} x_1 \\ x_2 \end{matrix} \rightarrow \textcircled{S} \rightarrow y$		$y = x_1, \text{ if } x_2 \geq 0.5$ $y = 0 \text{ otherwise}$
Sense Indicator	$x \rightarrow \textcircled{SI} \rightarrow y$		$y = 1 \text{ if } x > 0$ $y = 0 \text{ otherwise}$
*Memory (MEM)	$x \rightarrow \textcircled{\text{MEM}} \rightarrow y$		$y = 0 \text{ when first powered "on." If } x' \text{ is the first occurrence of } x > 0, y = x' \text{ thereafter.}$
*Transmittance (T)	$x \rightarrow T_{(n)} \rightarrow y$	--	$y = 0 \text{ initially. If } x' \text{ is the occurrence of } x > 0. \text{ Thereafter, } y = \min(1, i/n) \cdot x', \text{ where } m = \text{no. of times the input value } x' \text{ has repeated itself.}$

*See the following section for further explanation.

4.2 The Memory Elements

1. MEM-element

The MEM-element is initially "unset" (no value stored) and its output is zero. It "sets" to (stores) and begins to output its first nonzero input. The output of a "set" MEM-element is invariant and constant. At any point in time the output is the stored value. Subsequent inputs will neither change the stored value nor affect the output.

2. T-element

The T-element is also initially "unset" (i.e. no number stored) and its output zero. The T-element sets to the first nonzero input and begins to output the product $\frac{1}{n}$. (stored value) in a constant manner. The "stepping parameter" n is a nonzero positive integer. The next time the T-element encounters a nonzero input close to the stored value (additional logic insures closeness), a stepping process occurs by which the T-element's output is stepped to $(\frac{2}{n})$. (stored value). This process of stepping by adding $\frac{1}{n}$ to the fraction multiplier $\frac{m}{n}$ occurs each time a value the same as the stored value is input to the T-element until the term $\frac{m}{n} = 1$. At this point the T-element outputs the full value of its currently stored contents and can be considered to have reached "full potential". Any subsequent inputs to this element will not cause the stepping process to occur or affect the stored value. At any point in time after a T-element has reached full potential its output will be its stored value.

The particular value of its stepping parameter, n , is an inherent invariant characteristic of a T-element. In FROG, it is useful to have three variations of T-elements, having stepping parameters of 4, 2 and 1. This results in FROG having T-elements that come up to full potential quickly ($n = 1$), slowly ($n = 4$), or at an intermediate speed ($n = 2$).

3. T*-element

The principal macroelement of FROG's memory is a T*-element. It corresponds to the unit memory cell referred to in Section 2.4. A T*-element, shown schematically in Figure 4.1, is constructed out of the nine continuous logic elements described in Section 4.1 and some delay elements.

A T*-element has a maximum of three inputs (A_1 , A_2 to A_3) for the sensory (visual) signals of the stimuli and an input B for a reward (nonvisual) signal. A T*-element stores a set of values of a specific combination of the three visual characteristics taken one, two or three at a time. It also stores a value of nonvisual characteristic of one kind (e.g. a bad taste value) associated with the particular combination of values of visual characteristics stored. The T*-element forms a product of the maximum of the visual input values stored with the nonvisual value stored. The output of the element depends on how many occurrences of the given combination of visual inputs and the nonvisual input FROG has come across and will be described later.

Another input to a T*-element is a write regulating signal C. The element has two outputs: Output D is the main output of the T*-element and output E is a modification of input C.

The T*-elements are connected in a row. Figure 4.2 shows three of them connected together. Each of these elements retain a specific combination of values of two visual inputs. The visual inputs and the input B are the same for all the elements. However, input C, the regulating input NBHOOD, is set at 1.0 for the T*-element #1 (the leftmost T*-element) while input C for T*-element #2 is output E (the modified NBHOOD) of #1. The pattern is repeated.

To understand the operation of a T*-element consider a T*-element which is the leftmost one in the 'chain' described above.

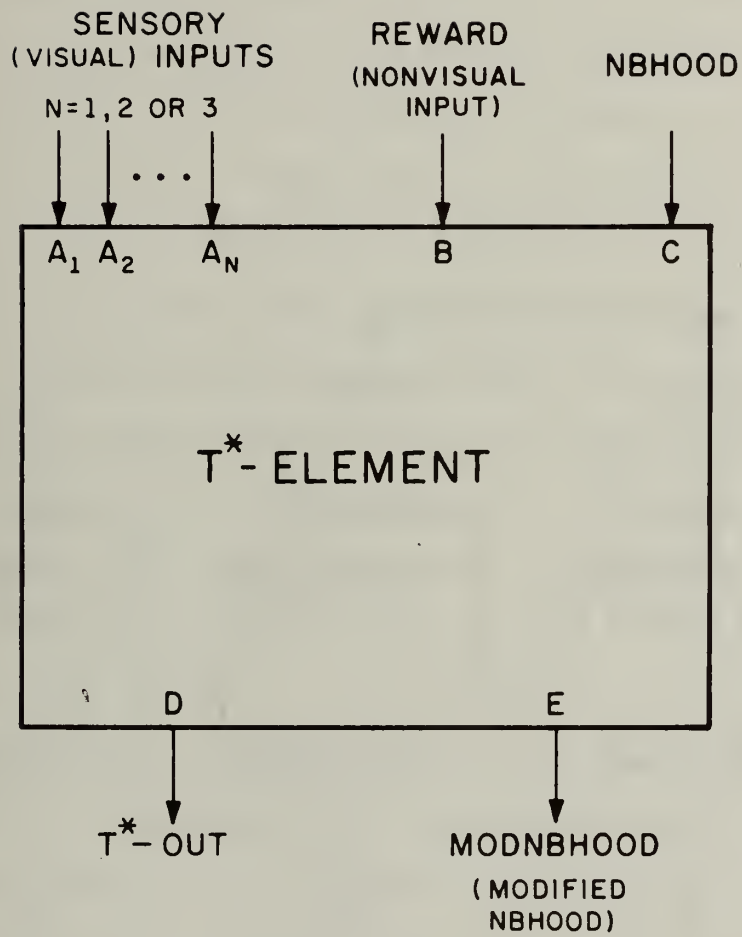


Figure 4.1 A T*-Element (Schematic)

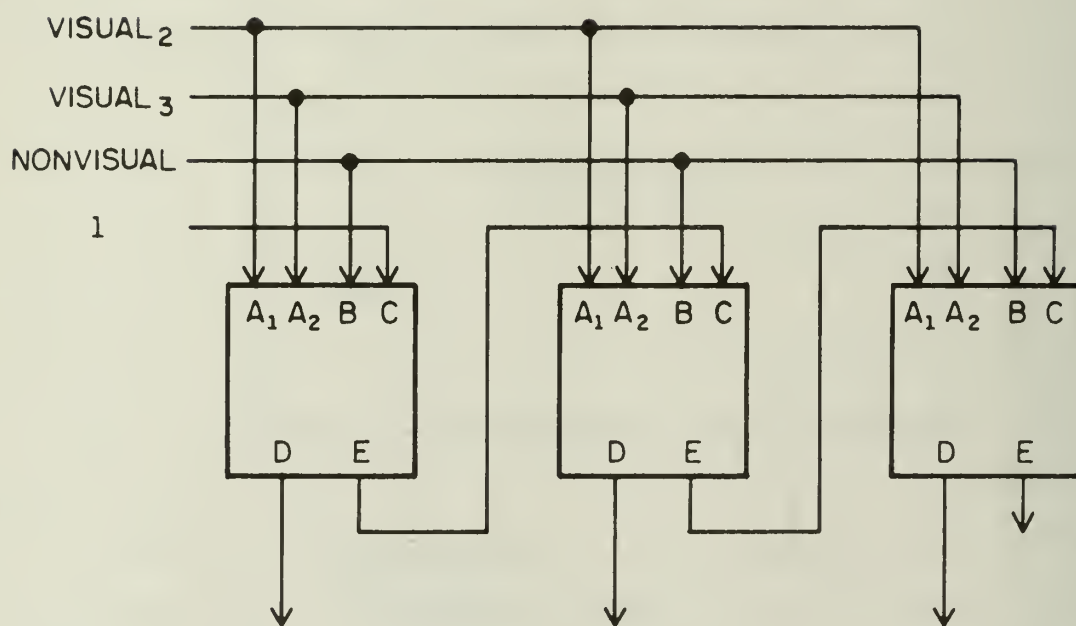


Figure 4.2 A Typical Chain of T*-Elements

Assume FROG has just "eaten" a bad tasting stimulus. The stimulus has specific values for its visual characteristics and bad taste. Assume FROG wants to record the specific combination of the values of the second and the third visual inputs (visual_2 and visual_3). The value of the second visual input appear on the A_1 line and that of the third visual input appear on the A_2 line. The value of the bad taste is placed on the B input line of the chain. Now, refer to Figure 4.3 for the following discussion. Consider what happens in block 1. If the B input value is nonzero, a 1.0 is input to the m-element of block 1. Another input to this m-element is the signal NBHOOD, which has been set initially at 1.0. The third input to this m-element (the complement of the output of the now "unset" MEM-element in Block 3) is also 1.0. The output of this m-element (which is 1.0) becomes an input to the M-element of block 1; the other input (the output of the still "unset" MEM-element of block 3) is 0.0. The output, then, of the M-element is the same value (1.0) that was output from the m-element. This output, which is $\geq .5$, gates in the visual inputs through the S-elements in block 2 and the taste input through the S-elements in block 3. These stimuli then "set" the MEM-elements in their respective blocks. The MEM-elements, after being "set", begin to output the values of visual and nonvisual signals that are still being input to them. These values reach the E-elements, and since each of the elements has two equal inputs, they now output the values of visual inputs (in block 2) and the taste value (in block 3) that were input to the T*-element. The M-element in block 2 now select the maximum value of the visual inputs.

One input to the block 4 is the maximum of all the visual input values stored in the MEM values of block 2. The other input is the taste value stored in the MEM-element of block 3. Each of these inputs go to a SI-element.

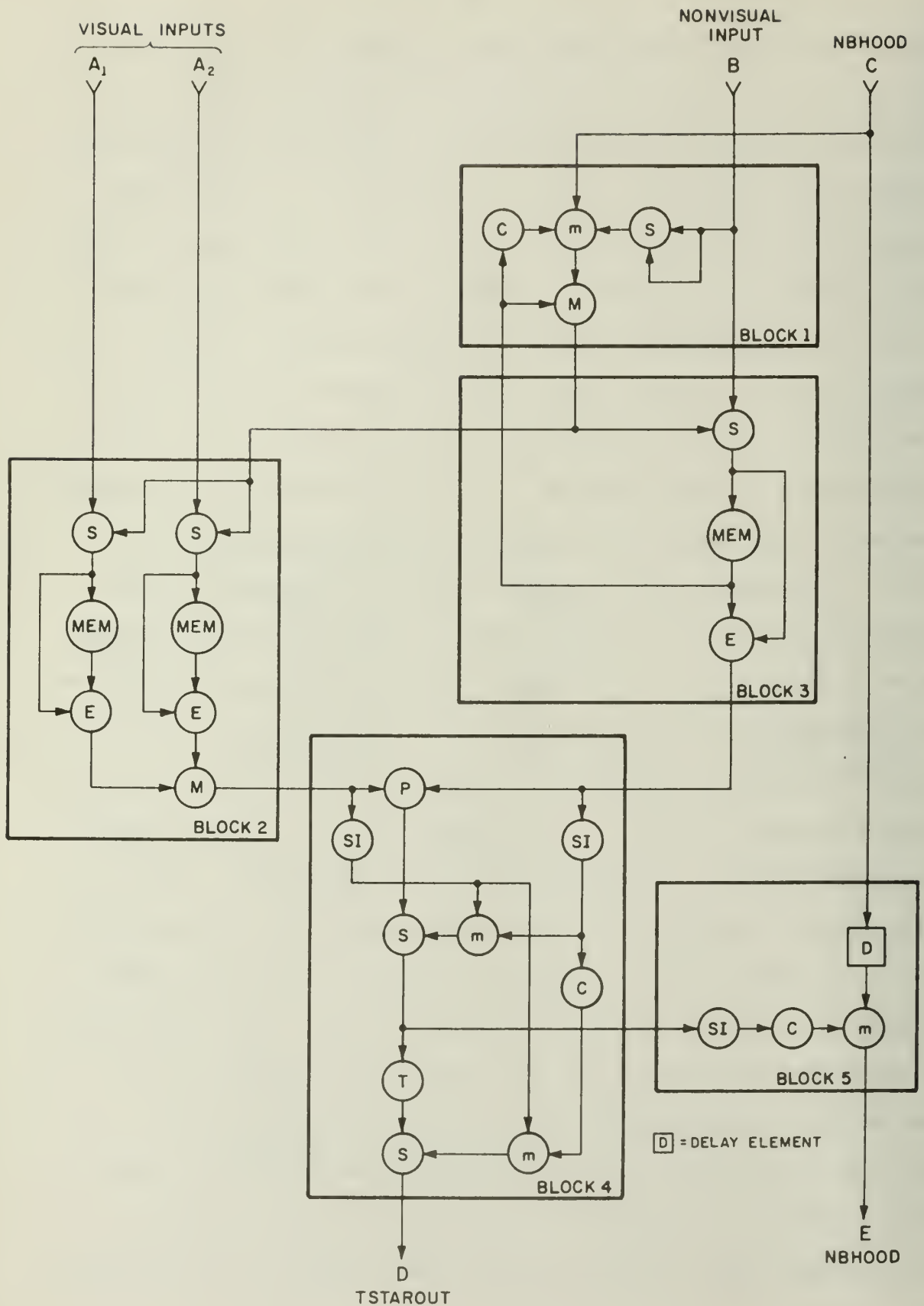


Figure 4.3 Logic Element Realization of a T*-Element

Since the inputs are nonzero the SI-elements begin to output 1.0. The inputs also go to a P-element which forms their product. The product is the x_1 input to the upper S-element in block 4. The x_2 input to this S-element is the minimum of the outputs of the above two SI-elements. Since these outputs are each 1.0, the x_2 input to the S-element is 1.0 and the output of the P-element is gated through to the T-element.

At this point, the E output of the T*-element, the modified version of the NBHOOD signal becomes defined. The E output of the T*-element is 1.0 only if in block 5 the delayed C input to the element is 1.0 and there is no output from the upper S-element of block 4. The 1.0 on the C input (NBHOOD signal) is delayed for long enough span, μ , to allow a signal to be output from the upper S-element of block 4 (i.e. to allow it to become reliable). Until this happens, since the upper input to the m-element in block 5 is zero, the E output is zero. Now, when the upper S-element of block 4 has an output value > 0 , the side input to the m-element in block 5 becomes 0.0, so that even after the delayed NBHOOD signal reaches the top input of the m-element, the E output continues to remain zero.

To return to block 4, the output from the S-element has caused the T-element to "set," i.e. store this value and begin outputting $\frac{1}{n} \times$ this value. The T-element output becomes the x_1 input to the lower S-element in block 4. The x_2 input to this S-element is the minimum function of (1) the SI function of the visual input to block 4 which is 1.0 and (2) the complement of the SI function of the nonvisual input to block 4 which is 0.0. Thus the x_2 input to the lower S-element in block 4 is 0.0 with the result that the D output of the T*-element is 0.0, which is what it should be, for this is a "write" into the memory cell, not a "read" out of it.

This is a complete "write" cycle for T*-element #1, but now let us consider what happens to the still "unset" T*-element #2. Its A_1 , A_2 and B inputs are

the same as that for the first cell, but the C input, NBHOOD (before and after modification by T*-element #1) is now 0.0. The effect of this will be to cause zero to be output by the m-element in block 1 and since the MEM-element in block 3 is unset, the output of the M-element will be 0.0. This zero is the x_2 input for each of the S-elements in blocks 1 and 2 with the result that the visual and nonvisual inputs to the T*-element will not be gated into the T*-element. However, let us look at the output E of this second T*-element, for it will be input for the T*-element #3. The complement of the SI function of the upper S-element in block 4 will be 1.0. This value will be one input to the m-element in block 5; the other will be the value coming out of the delay unit. However, this value, both before and after delay time, μ , has elapsed, will be 0.0. Therefore, after one memory cell in the chain has been written into, no "unset" T*-element to the right of that memory cell may be written into.

Now consider what will happen if a different set of A_1 , A_2 input values are input to the chain. They will reach the T*-element #1, the one that is "set." As has been said previously, a "set" T*-element always gates the A and B inputs to block 2 and 3. Since the MEM-elements are already set, the new inputs will not affect them, but will be compared with the MEM-element outputs at the E-elements. Since we are assuming that at least one visual input value is not within allowed tolerances for the E-elements, one or more E-elements in block 2 will output zero. The visual output value from block 2 will then be zero. So both the S-elements in block 7 will be inhibited, and the D output of the leftmost T*-element will be zero. Also the E-output of this T*-element will be 1.0 after time μ . So input C of the T*-element #2 is 1.0 and this will become "set" in exactly the same manner as the leftmost T*-element.

The process by which a T*-element is written into a second or third time is very much like that by which an "unset" T*-element is written into. Recall that only in the case in which all E-elements of blocks 2 and 3 output a value, will an input reach the T-element. Thus, only in repeated occurrences of the values stored in the MEM-elements (or values within the 0.05 neighborhood of these stored values) will the T-element go through the stepping process.

The NBHOOD signal functions solely to tell the chain that a T*-element has been written into for a given combination of visual inputs and, hence, once this occurs, not to write into it for a different set of visual inputs. This signal also makes sure that the writing of a new combination proceeds sequentially down the chain from left to right.

All of the above discussion pertains only to "writing" into the T*-element memory cells. Now the question arises how is a T*-element caused to output the current output state of its T-element? The process occurs contextually, when FROG has just seen a stimulus and its memory cells must be consulted to see if FROG has had any experiences with this stimulus in its past and if so, whether it was a good or a bad experience. Obviously, no nonvisual input is presented to the memory in this case, for feeding has not yet taken place; thus, the nonvisual input to a T*-element during a read cycle is zero.

The "reading" process occurs in the following manner. The visual inputs are presented to all T*-element in the chain at once (reading, it will be seen, does not have to occur sequentially, as does writing). The visual inputs will be gated into block 2 of each of the "set" T*-element in the chain. If the current visual input values are close enough (to within ± 0.05) to that which are stored in the block 2 MEM-elements, then the E-elements in block 2 will output nonzero numbers; if not, no read will occur. Now recall that the

x_2 input to the lower S-element in block 4 is the minimum of two inputs (1) the SI function of the visual output from block 2, which is 1.0 and (2) the complement of the SI function of the block 3 E-element output, which (since the B input to the T*-element is zero) is 1.0, with the result that this second input to the m-element is also 1.0. Thus, since the m-element is outputting 1.0, the current output status of the T-element is caused to be output; no stepping of the T-element will occur for there is no input to it in the read cycle.

A write into an "unset" element cannot possibly occur by accident during a read cycle, for if a T*-element is "unset" and the nonvisual input is zero, then the visual stimulus input will not get past the S-elements in block 2.

5. FROG: SYSTEM DESIGN

5.1 Structure of the Memory

The structure of the MEMORY is highly modular. The principal building block of the MEMORY is a T*-element.

The MEMORY is divided into three sub-memories (see Figure 5.1). The first is the GOOD TASTE MEMORY (GTM), which associates the distinguishing sensory (visual) input values of various stimuli with their good taste values. The second is the BAD TASTE MEMORY (BTM), which associates the visual feature values of bad tasting stimuli with their taste values. The third is a PAIN MEMORY (PM), which associates visual features of dangerous stimuli with the amount of pain they inflict. The nonvisual (reward) inputs to the GTM, the BTM and the PM are the good taste value, the bad taste value and the pain value respectively. The three visual inputs are received by each of these.

Each of GTM, BTM and PM is again divided into three levels. Level 1 associates the nonvisual input values with the three basic visual features of the stimuli taken individually. Levels 2 and 3 associate the nonvisual inputs with the combinations of three visual features taken two and all three at a time, respectively.

The level 1 of each memory has three chains of T*-elements. The T*-elements in these chains are identical. The T*-elements of the first chain in GTM remembers the relation between various values of S_1 (differing by at least ± 0.05) and the corresponding good taste values. The T*-elements the second and the third chain do the same with S_2 and S_3 respectively. Similar operations are done by level 1 chains in the BTM and the PM.

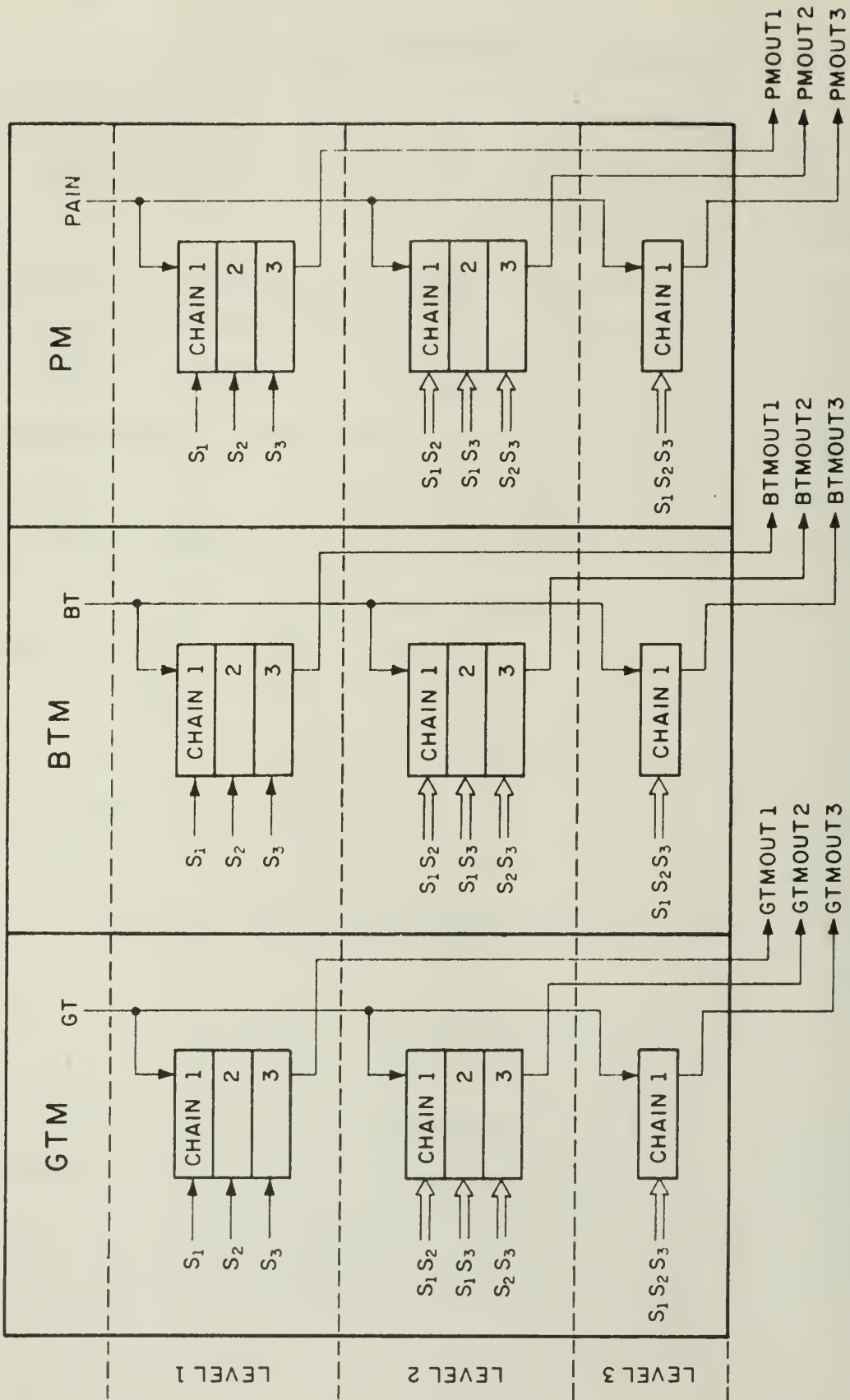


Figure 5.1 Structure of MEMORY

The level 2 of each memory has three more chains of T*-elements. The level 2 T*-elements are also identical to each other but differ slightly from the level 1 T*-elements. Each chain remembers unique values of a given combination of two different visual features and the corresponding GT (or BT or PAIN) values. The first chain receives S_1 and S_2 , so that unique combinations of S_1 and S_2 are remembered. The second chain receives S_1 and S_3 , and the third S_2 and S_3 .

The level 3 of each memory has one chain of T*-elements. Each element stores and correlates a unique set of values of three visual inputs and a nonvisual input. The different combinations of S_1 , S_2 and S_3 values and the corresponding nonvisual input values are remembered in the level 3 of GTM, BTM and PM.

A T*-element is, therefore, a relational memory between some elementary volumes of the sensory space and the reward space. Each T*-element is related to a node of behavior by reasons of its physical location in the MEMORY.

When a stimulus is presented to FROG, only the sensory (visual) inputs are sent to the MEMORY for a read (recognition cycle). All "set" T*-elements whose stored visual input value(s) fall within 0.05 of the given input values, will output the current correlation value stored in its internal Transmittance (T) element. A T*-element for which the input value(s) do not match the stored value(s), or for an element which is still "unset", the output is zero.

The maximum of the outputs of all T*-elements in each level of each memory is regarded as the level output. Figure 5.2 shows the organization of the T*-elements within a typical chain and the interconnections of the chain outputs to form the level output. Thus the level outputs from each of GTM, BTM and PM are the maximum of the T-element outputs in the individual

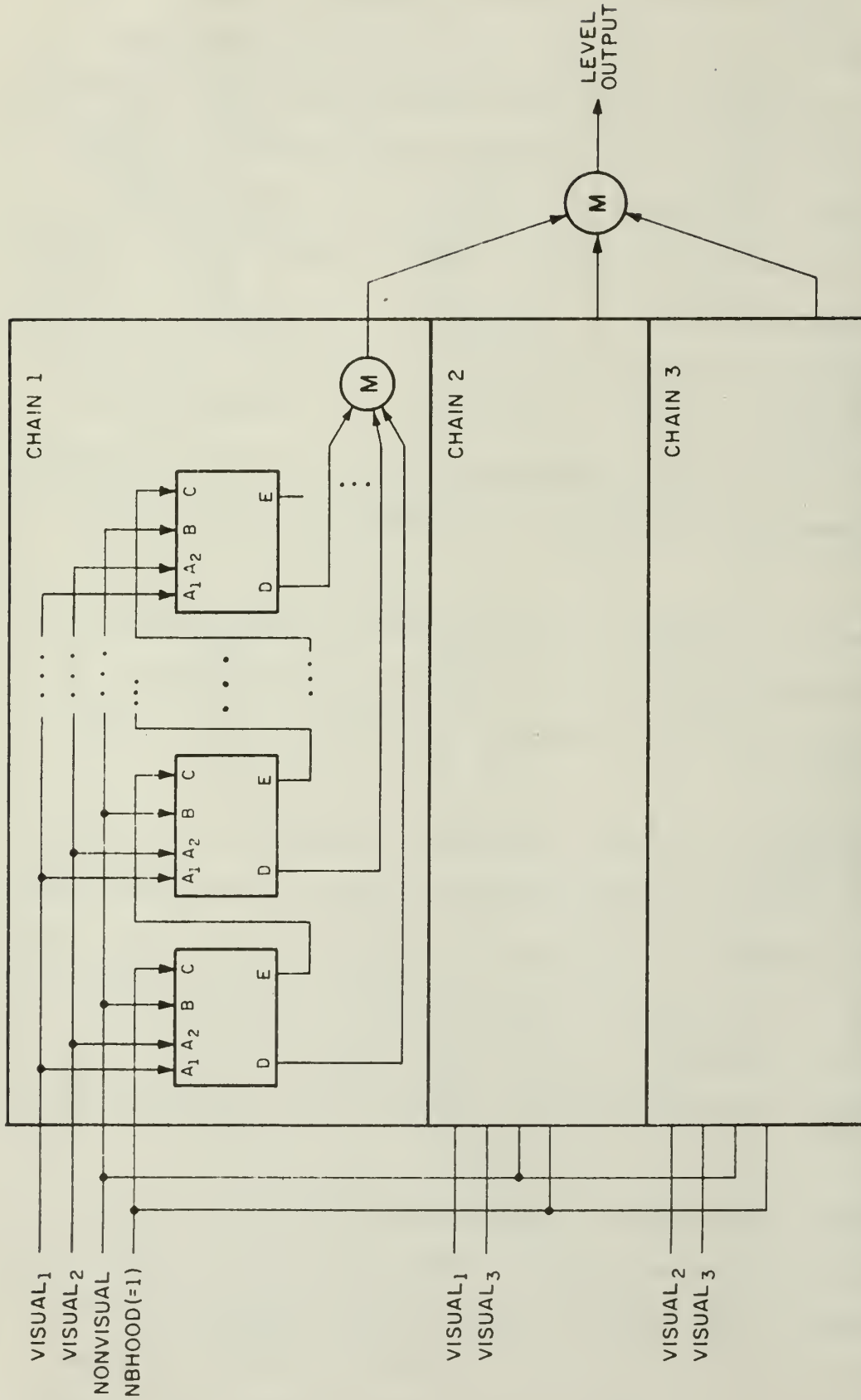


Figure 5.2 A Level 2 of a Memory with Three Chains of T*-elements

levels. The nine outputs are labelled GTMOUT1 - GTMOUT3, BTMOUT1 - BTMOUT3 and PMOUT1 - PMOUT3.

In the level 1, the T*-elements contain T-elements which have $n = 4$, so three repetitions are necessary for full learning. In level 2 $n = 2$, so one repetition is sufficient for the correlation value to achieve full potential. In level 3, the most specific level in FROG, $n = 1$, thus learning is immediate.

5.2 Memory Integration Unit

During a recognition (read) cycle the visual feature values are sent to the MEMORY. The three level outputs from each of GTM, BTM and PM are then handled by the MEMORY INTEGRATION UNIT (MIU). The MIU compares three signals at a time, starting with the three level 1 outputs. This comparison is done in an element called the MEMORY INTEGRATION SUBUNIT (MIS) described later. The outputs from the MIU are ALARMOUT, MIUOUT and NODECF (see Figure 5.3). If FROG senses a prey (good or bad tasting) there will be an output, MIUOUT, proportional to its desire to eat. But if the memory outputs are too feeble or too confusing for a meaningful response, the NODECF output become 1.0.

The logic realization of the MIS is shown in Figure 5.4. There are three inputs to the MIS; the three level outputs from the levels of GTM (or BTM, or PM). There are three outputs: MISOUT, ALARMIS and NODEC. if comparison of the input values result in a response (the response criteria are explained below) then either MISOUT or ALARMIS has a nonzero output; if no response can be generated, these outputs are zero. The third output, NODEC, tells the MIU whether there is a response or not; if a decision has been made, this output is zero, if not, it is one.

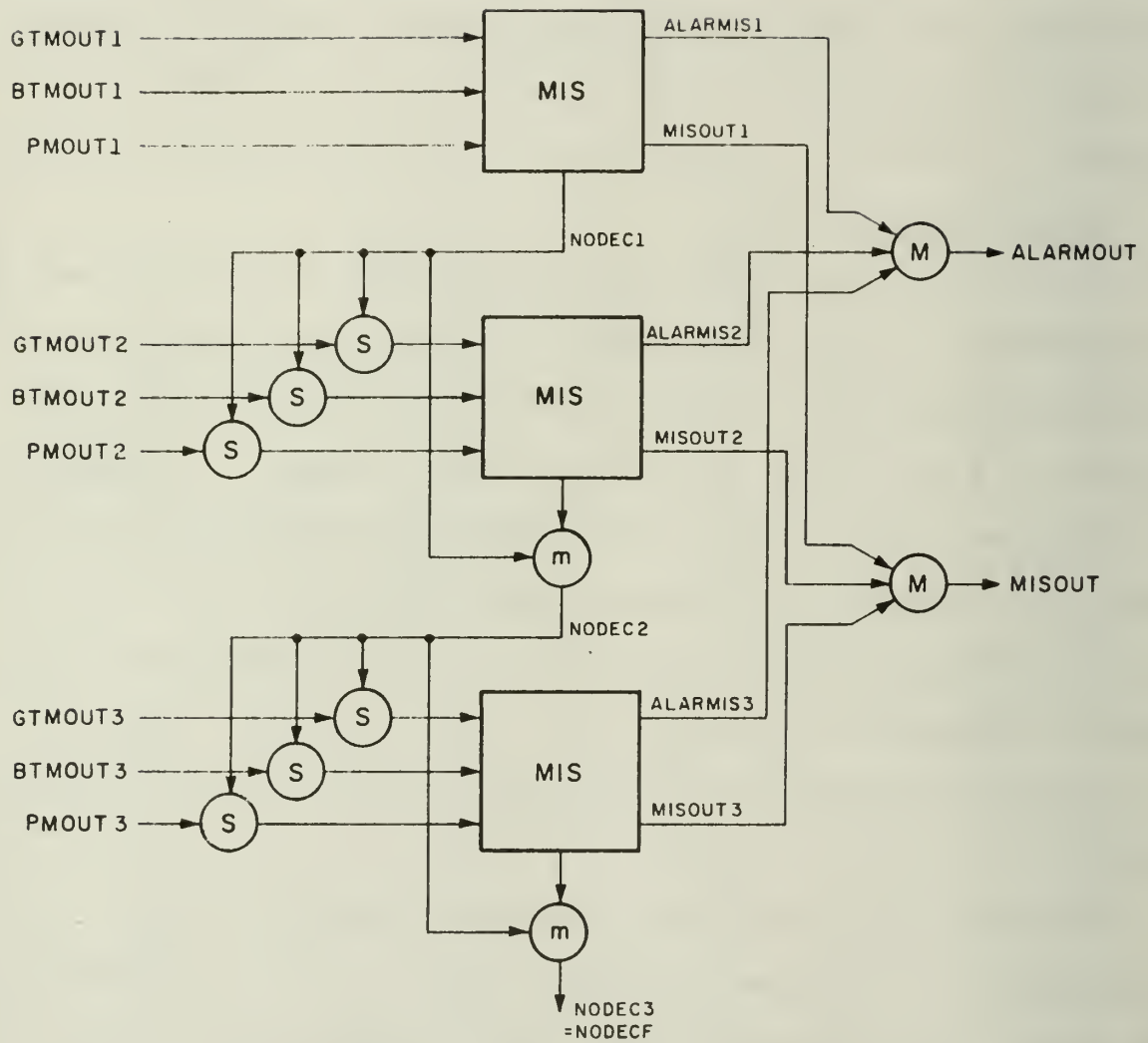


Figure 5.3 Block Diagram of a MIU

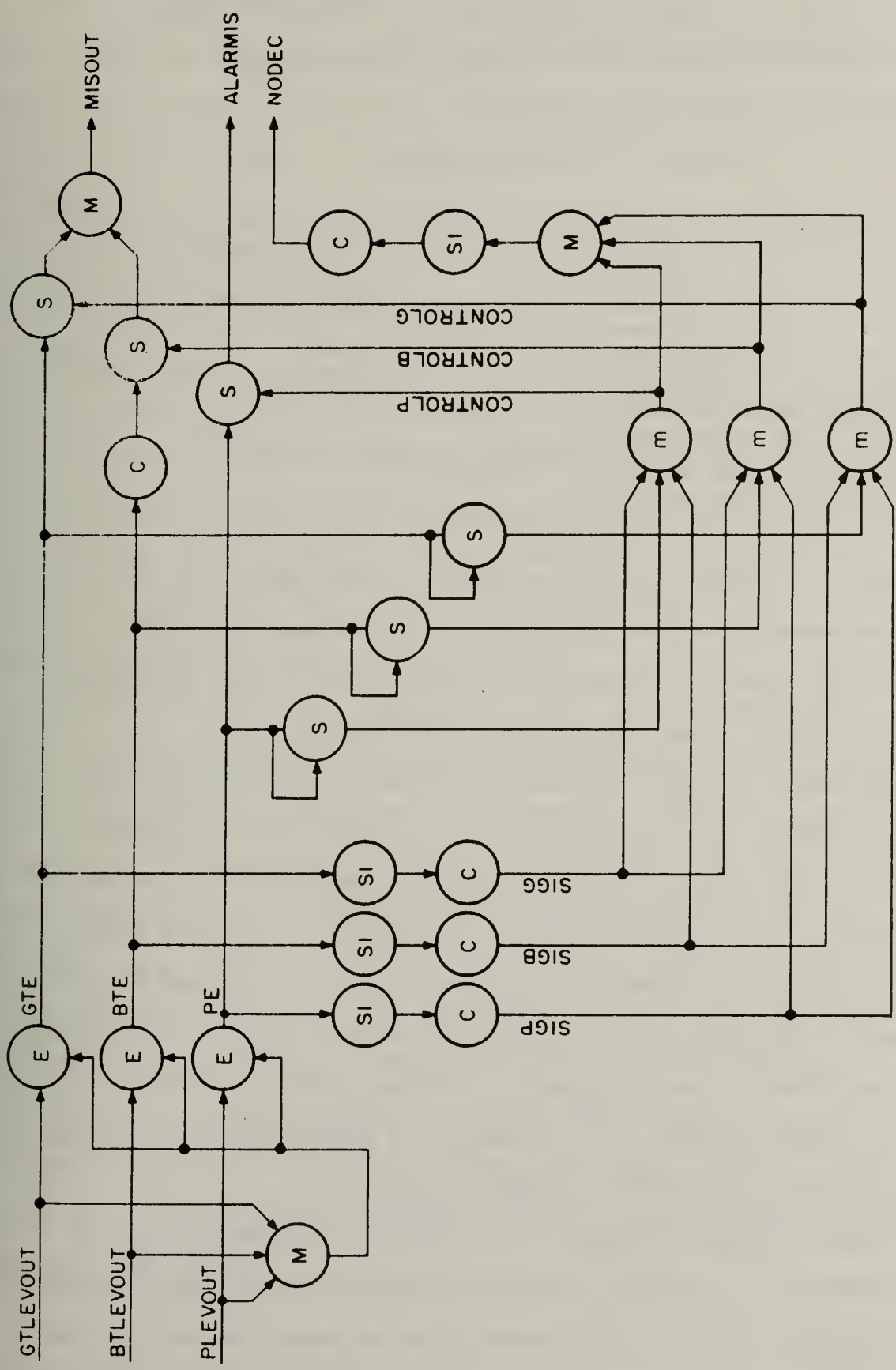


Figure 5.4 Logic Element Realization of a MIS

The criteria for a response from the MIS are detailed in Table 5.1. An input is called the largest if it is larger in value than that of all the other inputs by at least 0.05.

Table 5.1. Criteria for Response by a MIS

Condition	MIS OUTPUTS		
	MISOUT	ALARMIS	NODEC
GRMOUT is the largest and ≥ 0.5	GTMOUT	0	0
BTMOUT is the largest and ≥ 0.5	BTMOUT	0	0
PMOUT is the largest and ≥ 0.5	0	PMOUT	0
None of the input is largest by definition and/or all of them are < 0.5	0	0	1

Thus it is necessary for the numerically largest input to be greater than 0.5 and greater than the next input value by at least 0.05, if a decision is to be made. If these conditions are not met, then the outputs MISOUT and ALARMIS are zero and the output NODEC is one.

A MIU consists of three MIS modules with some additional logic to connect the modules (see Figure 5.3). The first MIS compares the level 1 outputs of the three memories. If no response is generated, based on the above criteria, then the output MISOUT1 and ALARMOUT1 are zero and NODEC1 is one. This allows the level 2 outputs to be gated into a second MIS. If comparison in this MIS yields no response the outputs MISOUT2 and ALARMOUT2 are zero and NODEC2 is one. The process continues for each of the MIS modules in a similar manner. If no response is generated after comparison of level 3 outputs, the NODEC3 (= NODECF) is made 1.0.

The above discussion assumes that no response is generated in any of the MIS modules. If, however, a response is generated at the nth MIS module, then $NODEC_n$ will be 0.0. This, by means of the m-elements between each module,

will cause 0.0 to be the x_2 input for all of the S-elements of the yet unused modules, preventing (1) any further comparison and (2) the NODECF from becoming active (it remains 0.0). Thus there will be only one non-zero input to the two M-elements and this will be the output at either MIUOUT or ALARMOUT.

The comparison within the MIU is so sequenced that it proceeds from the most general level of learning to the most specific (level 1 to 3).

5.3 Decision Unit

The decision unit is the subunit of FROG that makes all the decisions. Inputs to it are (1) ALARMF (2) MIUOUTF (3) HUNGER (4) the three basic visual inputs (5) NODECF (6) LOCATION (the signal which tells FROG in which location the stimulus has appeared) and (7) POSITION (the signal which tells FROG its present position, i.e. the location in which it presently is). The outputs from it are the various decisions.

The decision logic is as shown in Figure 5.5. From the inputs we derive a signal called FEED. Whenever the MIU comparisons yield an output, NODECF is zero. So the second input to the left hand M-element is zero. The output of this element is therefore equal to MIUOUTF. But when the MIU comparisons are indecisive, NODECF is 1.0 and MIUOUTF and ALARMF is zero. Under this circumstance, the output of the M-element equals 1.0, if hunger is ≥ 0.5 , or zero, when hunger is < 0.5 . The output of the M-element is one of the input to the top m-element. The other inputs to it are HUNGER and the innate visual stimulus (INNATESTIM), which is simply the maximum function of the three visual characteristics. It serves in the early trials to tell FROG whether or not what it sees is a stimulus, before its memories have learned enough about the stimuli.

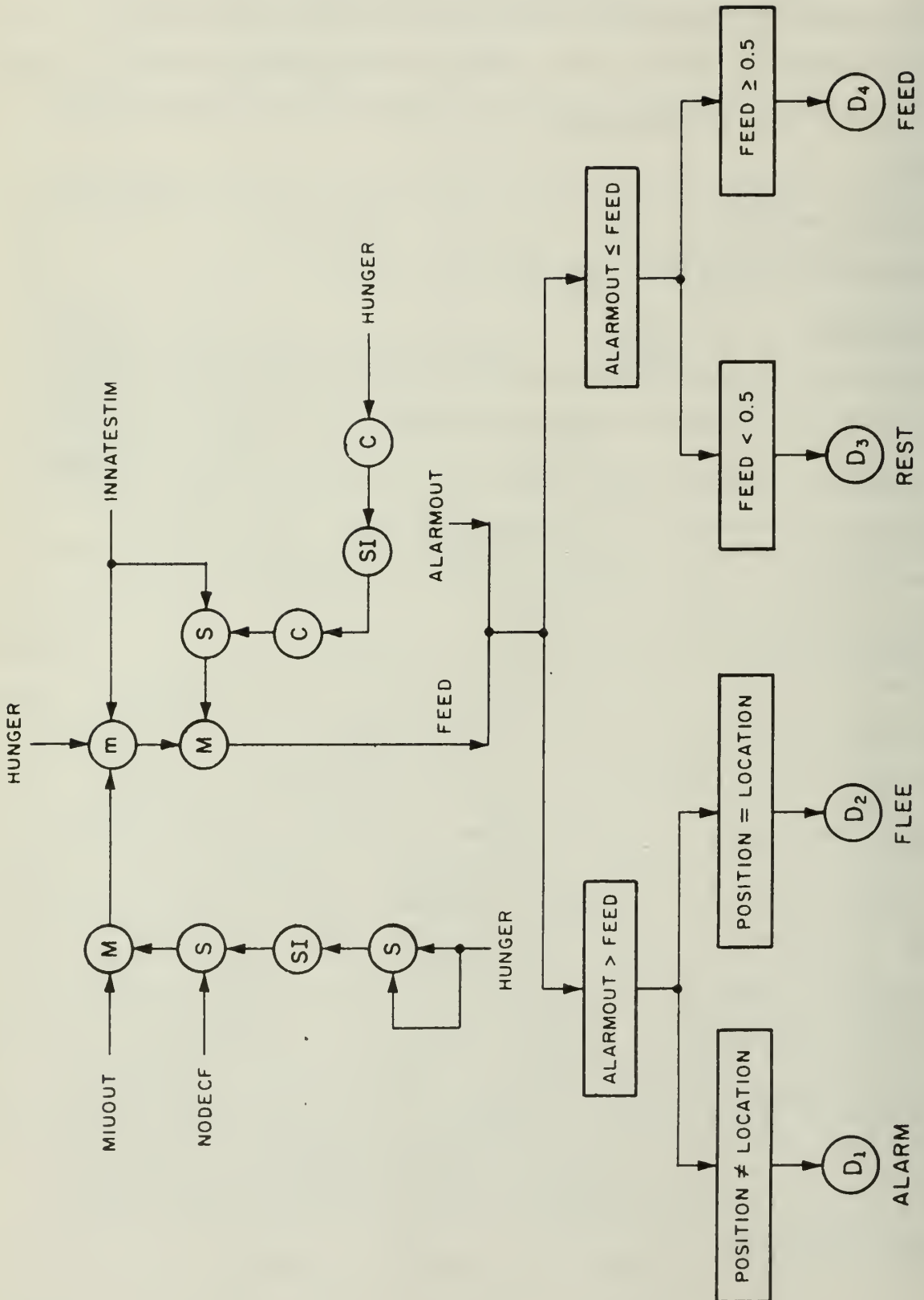


Figure 5.5 Decision Logic

The output of this m-element goes to a M-element. The other input to this M-element is nonzero only when hunger reaches the point of starvation (hunger = 1.0); when FROG is starving this input assumes a value equal to the innate visual stimulus. Thus, when FROG is at the point of starvation, its feeding impulse would be fairly large, i.e. FROG will eat any stimulus regardless of what its memories say about its good or bad taste. When FROG is not starving, the output of the m-element determines the value of FEED.

FROG can assume one of four decisions as detailed in Table 5.2. The tabulation is self-explanatory.

Table 5.2 The Decisions

Decision No.	Decision	Condition	Comments
1	ALARM	ALARMF>FEED POSITION≠LOCATION	FROG has seen a dangerous stimulus. It is ALARMed, but since it is in a different location, FROG decides to ignore the stimulus.
2	FLEE	ALARMF>FEED POSITION=LOCATION	The stimulus and FROG are in the same location. FROG has sensed a dangerous stimulus, so it decides to flee to its safe resting place.
3	REST	ALARMF<FEED FEED<0.5	FROG is not too eager to eat the stimulus.
4	FEED	ALARMF<FEED FEED>0.5	FROG decides to feed on the stimulus.

6. HARDWARE ORGANIZATION

6.1 Machine Representation of Numbers

The logic of FROG is designed in terms of a set of nine basic logic elements which operate in the signal continuum 0 to 1. One signal value can be distinguished from another by comparing them in an E-element (equivalence element). The E-elements can distinguish between two numbers which differ by at least 0.05 in their values. If, in the range 0 to 1, we choose a set of n values such that they differ from one another by at least 0.05 in value (i.e. are distinguishable) and any other number chosen in the range 0 to 1 is not distinguishable from one or more of the above set of numbers, then $10 < n < 20$. In other words, in the range 0 to 1, we can, in the limit, have a maximum of 20 distinguishable numbers, while the minimum is 10. So instead of working with a continuous range of values for the input variables we can quantize the input range to n levels. A variable x can assume one of n values depending on the level in which it falls. $n = 16$ is a convenient number so that the inputs can be coded into a 4 bit binary number.

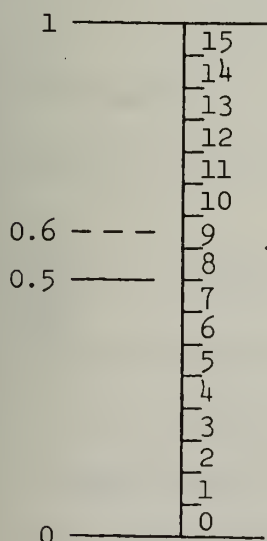


Figure 6.1

The machine representation of X of a variable x ($0 \leq x < 1$) is the binary equivalent of $\lfloor 16x \rfloor$, where $\lfloor y \rfloor$, the floor of y , is used to denote the integer satisfying $y - 1 < \lfloor y \rfloor \leq y$. For $x = 1$ the representation is the binary equivalent of 15 (=1111).

It can be easily shown that $1 - x$ is represented by the 1's complement (bit wise complement) of the representation of x . For example, if $x = 0.6$, then X = machine representation of x .

Or, $X = \text{binary equivalent of } \lfloor 16x \rfloor.$

Or, $X = 1001$ (see Figure 6.1).

and the machine representation of $1 - x = 0110$ [check: $\lfloor 16(1-x) \rfloor$
 $\lfloor 16 \times 0.40 \rfloor = \lfloor 6.4 \rfloor = 6 = 0110$ binary]. Notable exceptions occur when x is
on the boundary between two levels, i.e. x is a multiple of $1/16$ ($= 0.125$).
If, for example, $x = 3 \times 0.125 = 0.375$, then $X = \lfloor 16 \times 0.375 \rfloor = \lfloor 3 \rfloor = 3 = 0011$
and 1100 should represent $1 - x$. But $\lfloor 16(1-x) \rfloor = \lfloor 16 \times 0.675 \rfloor = \lfloor 13 \rfloor = 13 = 1101$.
Actually in any physical situation it is not possible for x to be right on the
boundary or the threshold value between two levels so $x = 0.375$ may actually
be $x = 0.375 + \Delta x$. Then $X = \lfloor 16x \rfloor = \lfloor 3 + 16\Delta x \rfloor = 3 = 0011$ and $\lfloor 16(1-x) \rfloor =$
 $\lfloor 16 - 3 - 16 \Delta x \rfloor = 12 = 1100$ which is the 1's complement of X .

6.2 Hardware Realization of the Basic Logic Elements

Having decided the number representation within the machine, the next
problem is to implement the special higher level logic elements in terms
of lower level elements, such as logic gates, registers, counters etc.,
and then to translate the system design into this directly implementable
(from a hardware point of view) logic.

Realization of some of the basic logic elements are shown in Figure 6.2.
This diagram is quite self explanatory. The realization of some of the other
elements are discussed below.

The Minimum (m) and the Maximum (M) Element

If $x_1, x_2 \dots x_k$ are the inputs to an m -element then the output, y , of
the element is defined by $y = \min_{i=1}^k (x_i)$. Since a number x_i [$0 \leq x_i < 1$]
is represented by a four bit fraction in the machine, a two-input m -element
can be easily realized by a 4 bit magnitude comparator (SN 7485) and a few
gates, as shown in Figure 6.3. The figure also shows how a k -input m -element
can be realized by cascading $k-1$ two-input m -elements.

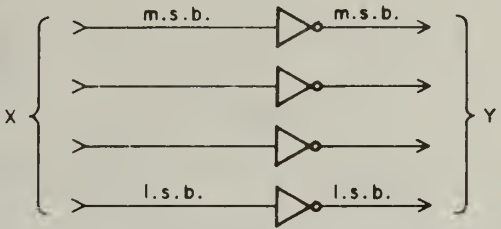
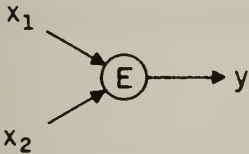
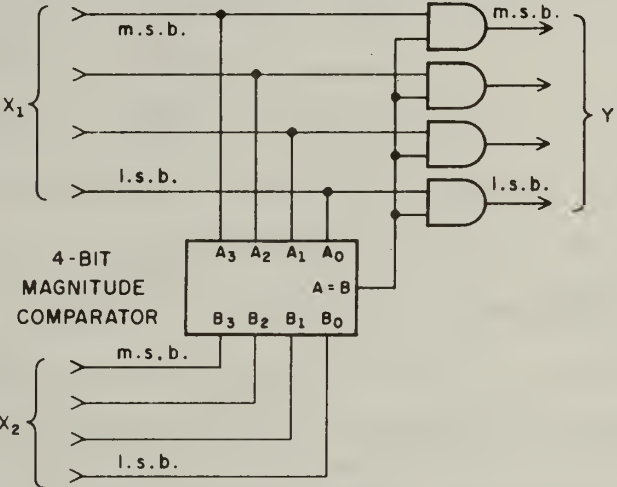
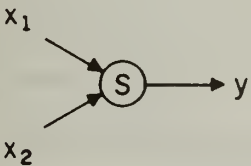
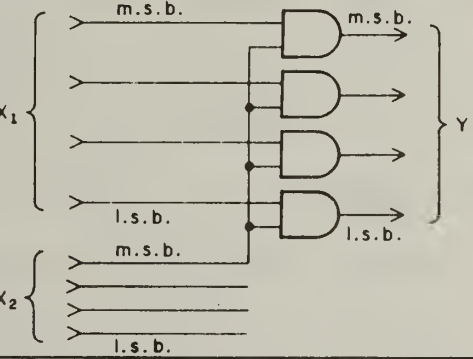
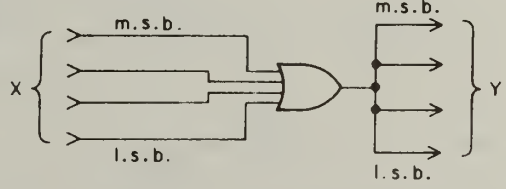
LOGIC ELEMENTS		
NAME	SYMBOL	REALIZATION
COMPLEMENT		
EQUIVALENCE		
SELECTIVE		
SENSE INDICATOR		

Figure 6.2 Realization of Some of the Basic Logic Elements

for FROG. (l.s.b. = least significant bit,

m.s.b. = most significant bit)

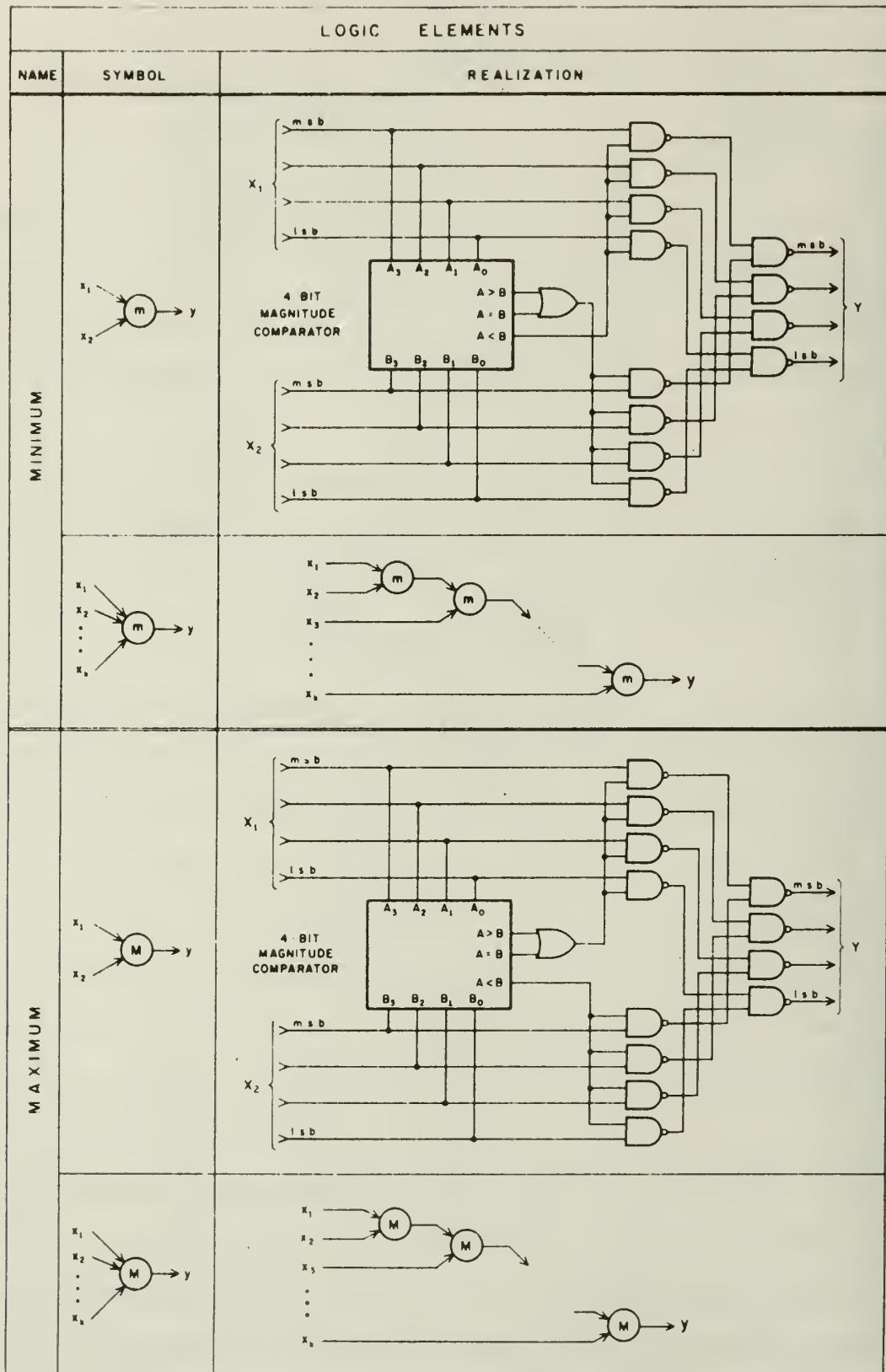


Figure 6.3 Realization of m- and M- Elements. (l.s.b. = least significant bit, m.s.b. = most significant bit).

The M-element is similarly defined. Figure 6.3 shows the realization of that too.

The Memory (MEM) Element

The MEM-element is initially "unset" (no value stored) and its output is zero. It "sets" to (store) and begins to output its first nonzero value. The output of a "set" MEM-element is invariant and constant. At any point in time the output will be the stored value. Subsequent inputs will neither change the stored value nor affect the output.

A hardware realization of this element is shown in Figure 6.4. Use is made of D-type bistable latches (SN 7475). Initially the circuit remains in a stable state with zero input, zero output (input and output lines all low) and the clock line high. The first time the input assumes a nonzero value, i.e. one or more of the input lines become nonzero, the output follows suit. This causes the clock line to go low (assuming the reset line to be low) and lock the input data into the latches. The output is now held steady and it remains unaffected by the input. The latches may be cleared by holding the input lines low (zero input) and making the reset line high. This makes the clock line high and puts the circuit back to the initial condition, whereupon the reset line can be made low again.

The Product (P) Element

If x_1 and x_2 are the inputs to a P-element, the output y of the element is the product, $x_1 x_2$, of the inputs.

The product of two 4-bit binary numbers x_1 and x_2 is easily obtained by a stochastic method. Figure 6.5 shows the block diagram of the P-element. The principle of operation is detailed below. Multiplication of two binary numbers by stochastic methods has been discussed in more details by Y. K. Wo in his doctoral thesis [10].

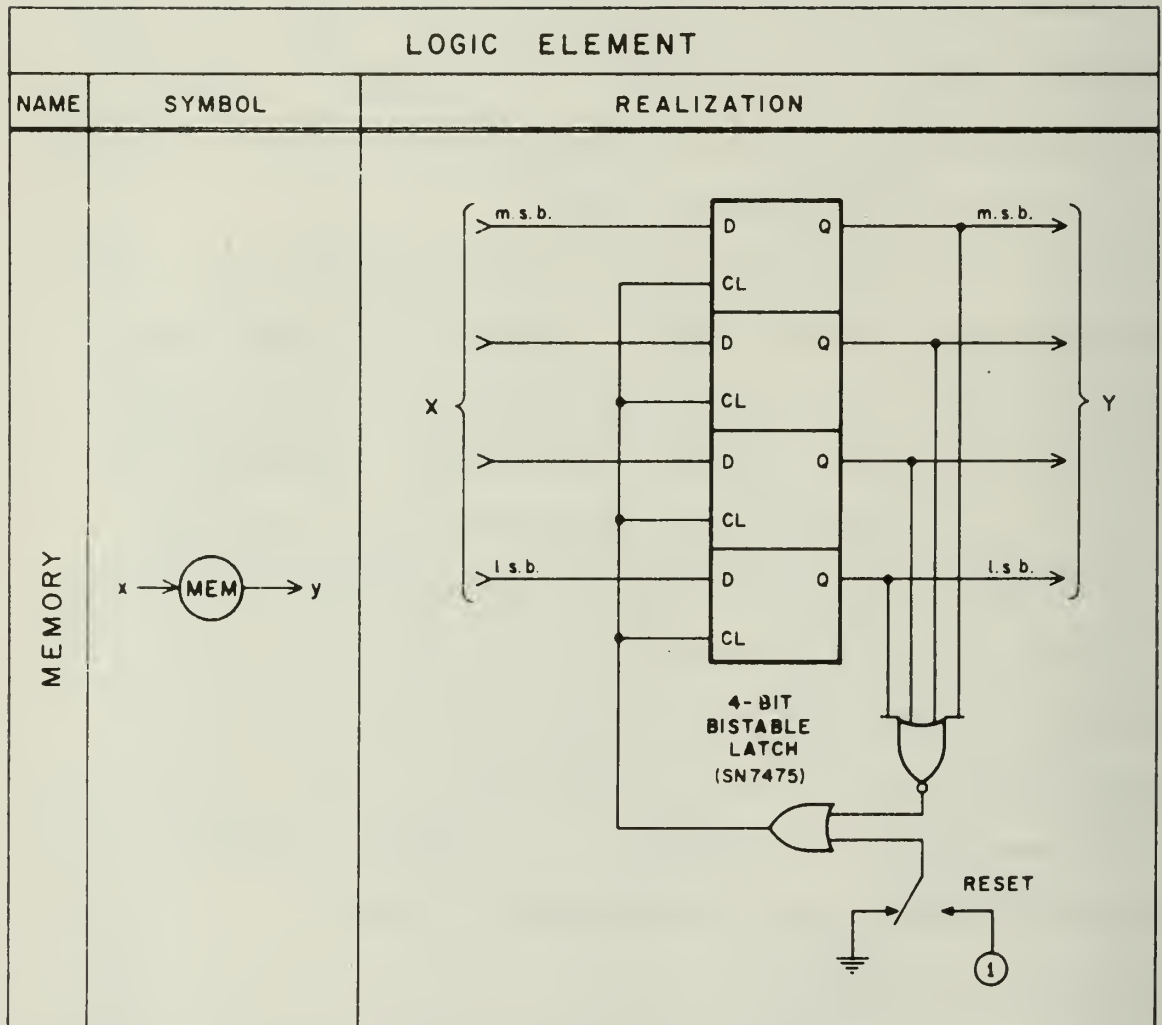


Figure 6.4 Realization of the MEM-Element

(l.s.b. = least significant bit

m.s.b. = most significant bit).

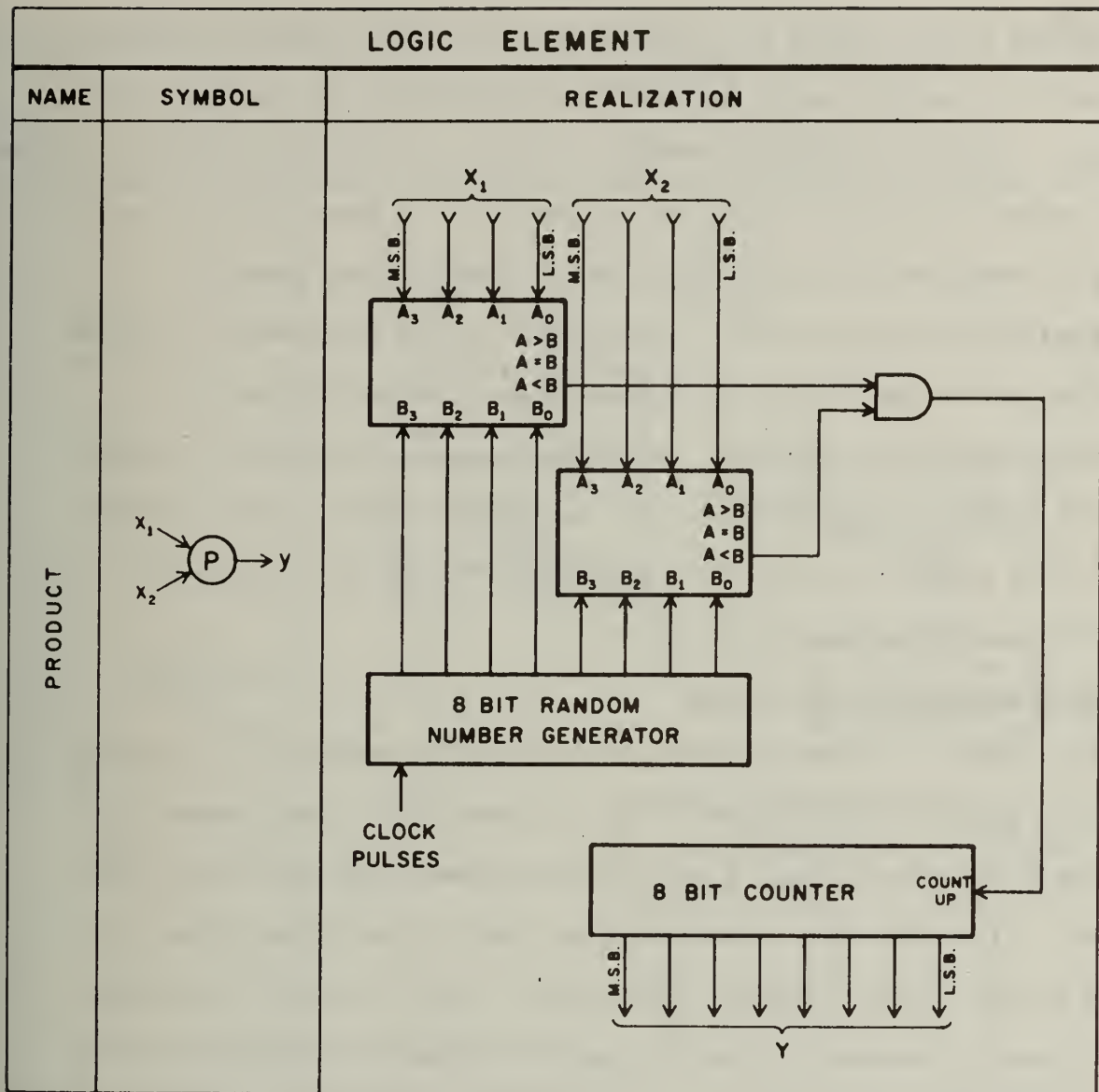


Figure 6.5 Realization of a P-element.

(l.s.b. = least significant bit,

m.s.b. = most significant bit)

The two binary numbers are compared in the two 4-bit magnitude comparators with two 4-bit random numbers that are derived from an 8-bit random number generator. For the generation of random numbers a pseudo-random 8-bit shift register sequence is used. As the generator is cycled through its states the comparator outputs produce two stochastic pulse sequences corresponding to the two numbers. The product of the two numbers is formed by ANDing the two sequences. The output pulses from the AND gate are accumulated in an 8-bit counter. If the random number generator is cycled through a suitable number of states the accumulated number in the counter is the 8-bit product Y of the two 4-bit numbers X_1 and X_2 . The counter needs to be set to all zeros initially.

The Transmittance (T) Element

A T-element is a memory element as described in Section 4.2. Note the quantities defined for the element. In the hardware for (see Figure 6.6), the circuit has a 4-bit input X and requires signal FRESH INPUT (FI) which receives a pulse everytime a nonzero input, same as the stored value, is applied to the circuit. For the first nonzero input, to which the element sets, a pulse is produced on the FI line, even though the stored value just prior to this input is a zero.

The signal FI is utilized to increment a special counter called the M-COUNTER which maintains the current value of M till $m = n$. After this the counter value is kept frozen at n . Thus the output of the counter at any instant is $z = \min(m, n)$.

The first time the input is nonzero, it is immediately locked into the 4-bit latch (in a manner similar to that in an MEM-element). The FI signal initiates a conversion of this input binary number to a corresponding synchronous random pulse sequence (SRPS). The method of conversion is the

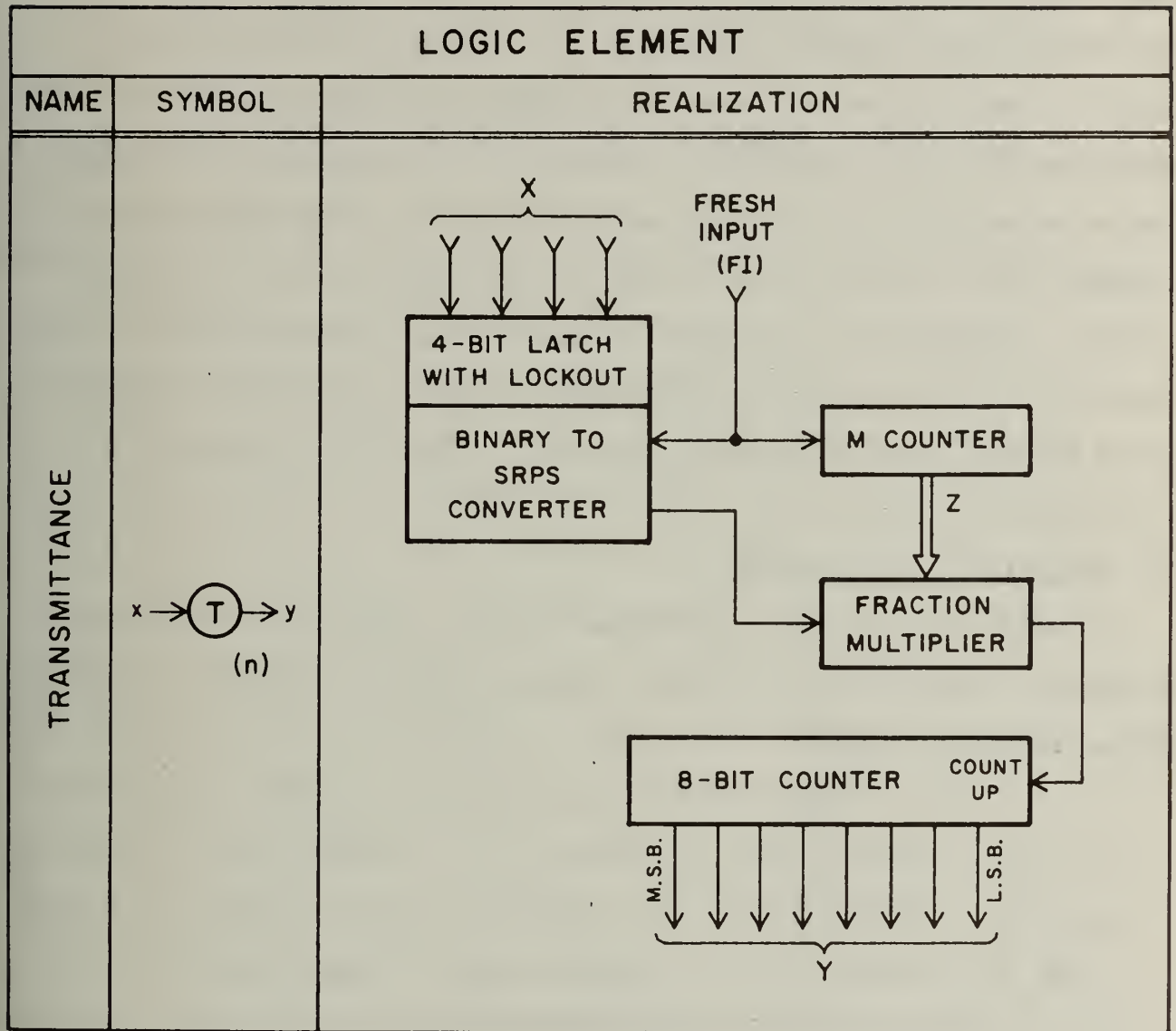


Figure 6.6 Realization of a Transmittance (T.)

Element (Schematic)

(l.s.b. = least significant bit,

m.s.b. = most significant bit.)

same as that used in a P-element. This SRPS is passed through a circuit called the FRACTION MULTIPLIER which suppresses $n - z$ pulses for every successive group of n input pulses ($z = \text{output of M-COUNTER}$) in order to achieve a multiplication by the fraction $z/n = \min(1, m/n)$. The output SRPS is converted back to an equivalent 8-bit binary number Y simply by accumulating the pulse sequence over a suitable length of time in an initially cleared 8-bit binary counter. Although, the stored value in the 4-bit latch stays unaltered in time, the output is recalculated using the current value of z everytime a pulse appears on the FI line. Thus the output of the circuit, stored in the counter, is $Y = \min(1, m/n) \times \text{stored value}$.

6.3 Realization of T*-Elements

In order to provide for six different stimuli, only forty-two T*-elements are needed. The distribution of these elements in the various levels of GTM, BTM and PM is as follows:

	GTM	BTM	PM	
level 1	6 (#00 - 05)	6 (#06 - 11)	6 (#12 - 17)	
level 2	6 (#18 - 23)	6 (#24 - 29)	6 (#30 - 35)	
level 3	2 (#36 , 37)	2 (#38 , 39)	2 (#40 , 41)	
	14	+ 14	+ 14	= 42 total

There are two T*-elements per chain in each of GTM, BTM and PM. The T*-elements in different levels have different numbers of sensory inputs and different stepping parameters (see Section 4.2). A level 1 T*-element has one sensory input and its stepping parameter n is 4. In a level 2 T*-element, the number of sensory inputs is two and $n = 2$. A level 3 T*-element has three sensory inputs and for it $n = 1$. Instead of building forty-two T*-elements in hardware, FROG's circuitry is designed with one generalized T*-element module which is used forty-two times during any MEMORY read or

write operation. Data relevant to the forty-two elements are held in banks of registers (semiconductor memory IC's). The T*-element module performs proper functions on appropriately selected data.

The circuit realization of the T*-elements is shown in Figures 6.7 through 6.10. The circuit 'T*-element part I' (Figure 6.8) shows the data memory for forty-two T*-elements, comprising of twelve 16 words x 4-bit word semiconductor memory ICs (SN 7489). These chips are organized as three rows of chips to give 48 rows of data words. The data for the forty-two T*-elements are stored in the word rows 0 through 41, and selected by a 7-bit address on the lines LMADDR1 - 4 and $\overline{\text{LME1}} - \overline{\text{LME3}}$. The left column of ICs stores the A_1 input values (DMEMIN1). The next column of ICs stores the A_2 input values (DMEMIN2). The third column with one IC holds the A_3 input values (DMEMIN3). The fourth column of ICs hold the nonvisual (B) input values (DMEMIN4). The rightmost column of ICs hold a 4-bit word for each T*-element. These are essentially some status bits. The first bit (DELSET) is to indicate if the T*-element is set or unset; DELSET = "0"¹ means that the T*-element is yet unset and DELSET = "1"² signifies that the element is set (written into). The next three bits (DM3 - DM1) specify how many times the corresponding element has been written into, i.e., the current value of the parameter m (Section 4.2). The 4-bit comparator ICs (SN 7486s) form the E-elements of blocks 2 and 3 of T*-elements. These compare the stored values with the inputs. The results of the comparisons (LEQ1 - LEQ4 signals) decide if the element should read out its output value during a read cycle, and if it should get written into during a write cycle. The logic for doing this is in the center of the circuit 'T*-element part II' (Figure 6.8), which generates the LWRITE and LREAD signals.

¹ "0" = logic zero.

² "1" = logic one.

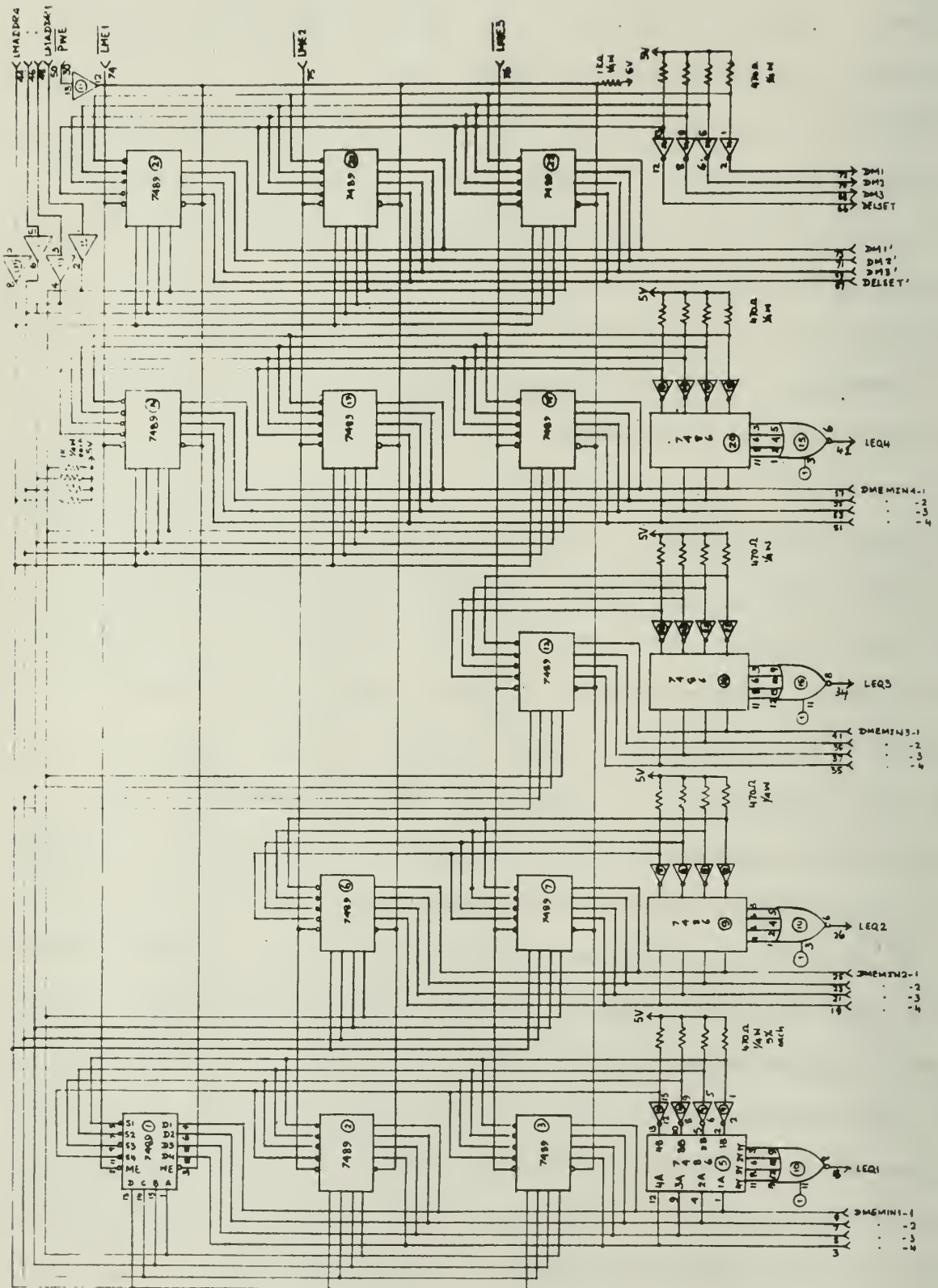


Figure 6.7 T*-Element Part I

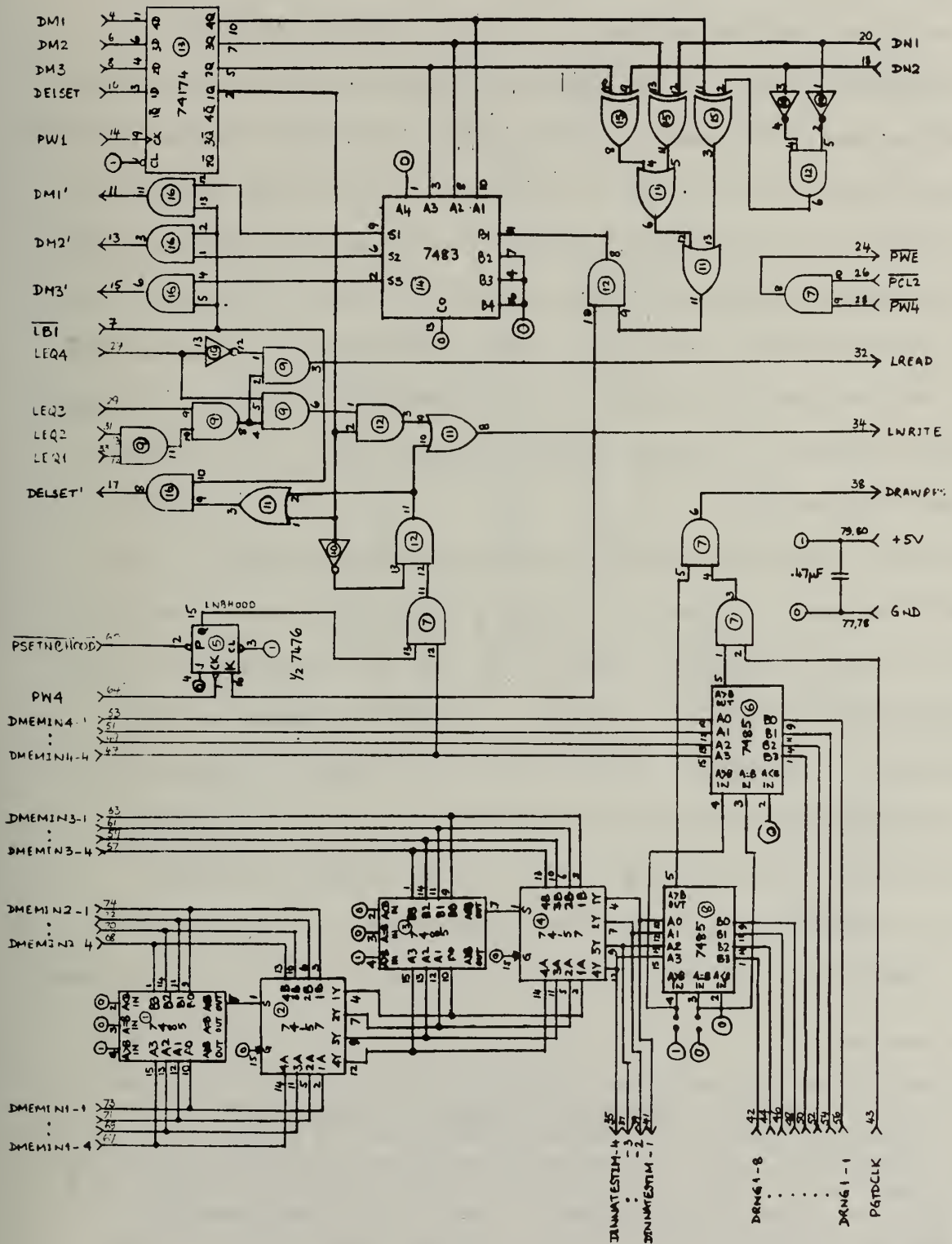


Figure 6.8 T*-Element Part II

The four ICs (numbered 1 through 4) on the left side of the circuit in Figure 6.8 generate the maximum of the sensory inputs. This is the INNATESTIM^1 value, when all the sensory inputs are applied to the circuit. When writing into a T*-element, a product has to be formed of the maximum of the applied inputs and the reward input value. To do that the two numbers are converted into two independent synchronous random pulse sequences (SRPS). The 8-bit pseudo-random number DRNG1, generated elsewhere in the system, is used for the conversion. The random number generator is cycled through 255 states. The two SRPSs are multiplied by the AND gates in the lower center of the circuit diagram. The output sequence (DRAWRPS) goes to the 'fraction multiplier' (Figure 6.9).

The 2-bit signal DN2 - DN1 in 'T*-Element part II' specifies if the selected T*-element is in level 1 or level 2 or level 3. Before a write cycle, the current status bits (DELSET, DM3, DM2, DM1) are loaded into the 4-bit latch SN74174 (IC #13). If DELSET = "0" and the element is being written into, or if DELSET is already a "1", then DELSET' is made equal to a "1". The number specified by DM is incremented by a 1 if the element is being written into and the number has not already reached the maximum value (i.e., already equal to the stepping parameter, n), and is returned as DM'. The four bits DELSET', DM3', DM2', DM1' are stored back in the memory

¹ This quantity has previously been referred to as the INNATESTIM signal in Section 5.3. In the actual hardware system, to all signal names another letter is added at the front to denote the nature of the signal. The convention followed is: A D at the beginning of a signal name signifies a data signal or data pulse train; a P signifies a timing signal; an L signifies a level status or a level control signal. Also a bar is drawn over the signal name if the signal is active low. A note should be taken of these facts when deciphering the for the signal names.

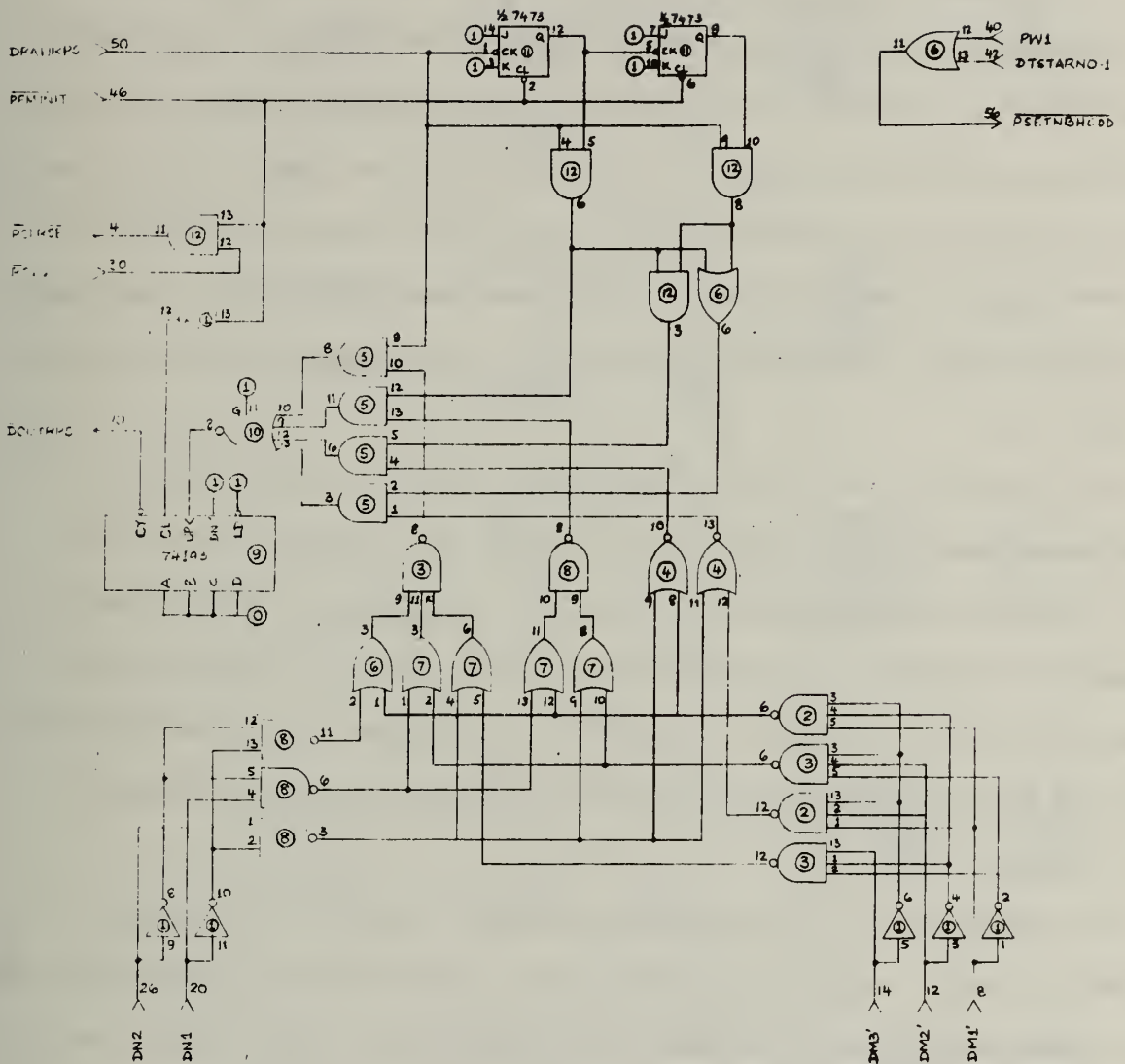


Figure 6.9 Fraction Multiplier

ICs as the new status bits, true after the current write cycle. All of this is performed by the logic on the left hand side of 'T*-element part II'.

The output of the flipflop at the center of the circuit is the LNBHOOD signal. During the write cycle of any trial, the NBHOOD flipflop is set if the T*-element addressed happens to be the first in the chain. This is determined by seeing if the DTSTARNO value is even. The flipflop is reset if this T*-element is written into. So, for the second T*-element in any chain, the LNBHOOD signal is 1.0 only if the first in the chain is not written into in that trial.

The purpose of the 'fraction multiplier' circuit (Figure 6.9) is to multiply the product from 'T*-element part II' by the fraction $\min(1, m/n)$. This is done by the suppression of an appropriate number of pulses from the product SRPS (DRAWRPS) and accumulating the rest of the pulses in an 8-bit binary counter (one for each T*-element). This number is then the correlation value for the T*-element and the counter serves as the inherent memory of the T-element inside.

The exact pattern of suppression, of course, depend on the values of DM' and DN bits which respectively specify m and n. The flipflops and the four logic gates immediately below the flipflops take the product SRPS (DRAWRPS) and generates four sequences with x pulses out of every group of four pulses suppressed, where $x = 0, 1, 2$ and 3 for the four sequences. This corresponds to multiplication by 1.00, 0.75, 0.50 and 0.25 respectively. The selection logic (AND - OR - INVERT gate) at the center of the diagram is set upto select the right pulse sequence. The selector is appropriately controlled by the logic at the bottom of the Figure, the inputs to which are DM' and DN bits.

In order to hold the correlation values of the forty-two T*-elements, the same number of 8-bit binary counters are needed. The correlation value is needed only in 4-bit precision for subsequent calculations. So, the last

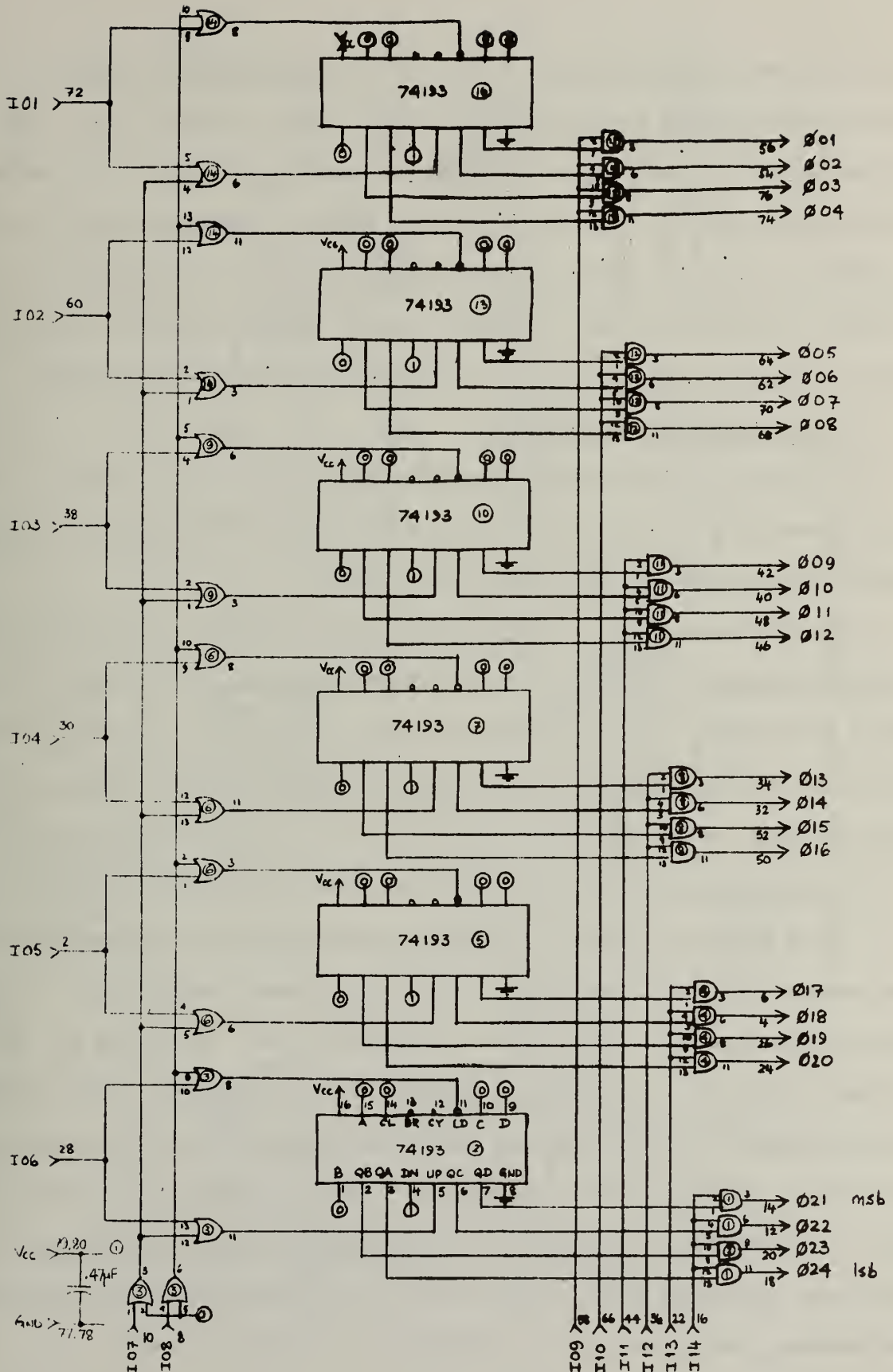


Figure 6.10 Counter Card

four bits need not be stored. This is effectively achieved by making the pulses accumulate in a 4-bit counter in the fraction multiplier and then by accumulating the carry pulses (DOU TRPS sequence) in a second 4-bit counter to give the top four bits of the correlation value. A separate counter is provided for each T*-element addressed. Forty-two 4-bit binary counters (forty-two SN74193s in seven 'counter' circuit boards, with six counters per board as shown in Figure 6.10) are provided and selected appropriately.

6.4 Overall Hardware Organization

The overall hardware organization of FROG is shown in Figure 6.11. Each box represents a circuit card in the actual set-up. The function of each of these circuit cards is described in this section.

The four circuit cards 'set up 1' through 'set up 4' have preset switches for programming-in the visual and nonvisual input values for the six different types of stimuli. When the stimulus number is specified, the signal DSTIMNO, the three sensory (visual) input values get defined on the DVISIN1, DVISIN2, DVISIN3 lines. Also the reward value (GT, BT or PAIN value) is made available on the appropriate one of the three outputs DGTSTIN, DBTSTIN, DPAININ.

As we mentioned in the last section, reading or writing from the T*-elements is sequential, because there is really one T*-element module which, in conjunction with relevant data, is used forty-two times. So during any read or write cycle, the bookkeeping is done by the use of a 6-bit binary counter called COUNTER2 in the 'register' card. The value of this 6-bit signal goes from the 'register' card to the 'decoder' card. The four least significant bits of DTSTARNO are used for the 4-bit LMADDR for the 'T*-element part I' card. The other three bits $\overline{\text{LME1}}$ - $\overline{\text{LME3}}$, going to 'T*-element part I', are generated by decoding the top two bits of DTSTARNO in the 'decoder' card.

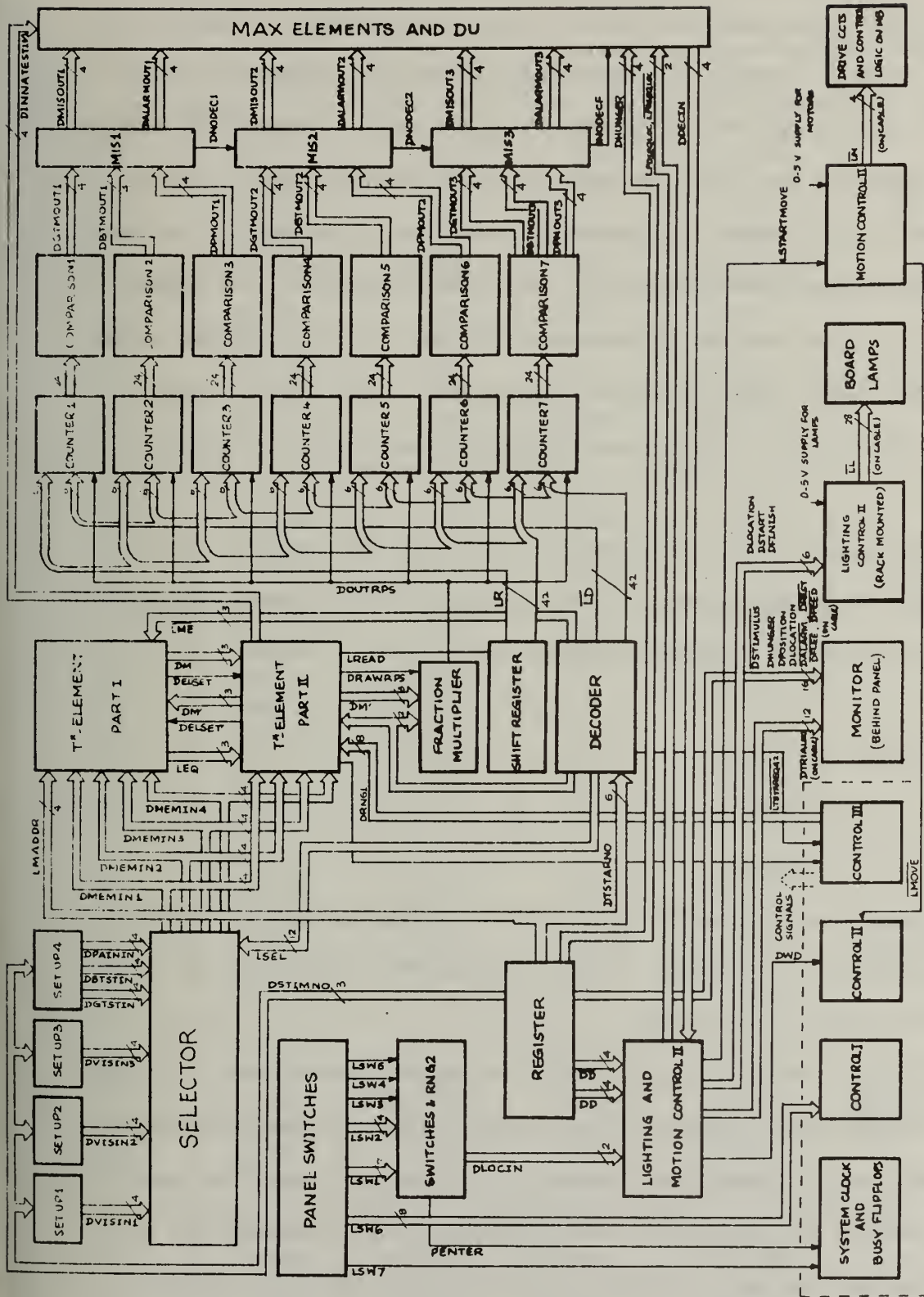


Figure 6.11 Overall Hardware Organization

The 'decoder' also generates twelve control signals LSEL1 - LSEL12 from the DTSTARNO signal. These go to the 'selector' circuit, the function of which will be explained shortly. Forty-two more control signals $\overline{LD00}$ through $\overline{LD41}$ are generated in the 'decoder' card from the DTSTARNO signal. These control signals go to enable the forty-two 4-bit binary counters in the seven 'counter' cards. When DTSTARNO value is zero, $\overline{LD00}$ is active, i.e., goes to "0", with all other control signals remaining at "1". When DSTARNO value is twenty-three, $\overline{LD23}$ is "0", the rest staying at "1" and so on. These signals enable the input to the proper counter during a write or a clear operation.

The fourth signal which is generated in the 'decoder' circuit is the 2-bit DN signal, which specifies the level number of the T*-element, being selected by the value of DTSTARNO signal in the following manner:

	DN2	DN1	Level #
$0 \leq \text{DTSTARNO} \leq 17$	1	0	1
$17 < \text{DTSTARNO} \leq 35$	0	1	2
$35 < \text{DTSTARNO} \leq 41$	0	0	3

This 2-bit signal also indirectly specify the stepping parameter n for the selected T*-element. This signal is used both in 'T*-element part II' card and in 'fraction multiplier' card.

The 'selector' card has as its inputs the signals coming from the 'set up' cards. It generates as its outputs, the data inputs to the 'T*-element' cards (DMEMIN1 through DMEMIN4). Obviously, the inputs to the 'T*-element' cards depend on the DTSTARNO value. The inputs for the various values of the DTSTARNO values are shown in Table 6.1. Necessary selection signals LSEL1 thorough LSEL12, derived from DTSTARNO in 'decoder', comes to 'selector' card to do the proper selection.

Table 6.1 Selector Circuit: Input-Output Relationship

DTSTARNO	DMEMIN1	DMEMIN2	DMEMIN3	DMEMIN4	
00	DVISIN1	0000	0000	DGTSTIN	level 1 GTM
01	DVISIN1	0000	0000	DGTSTIN	
02	DVISIN2	0000	0000	DGTSTIN	
03	DVISIN2	0000	0000	DGTSTIN	
04	DVISIN3	0000	0000	DGTSTIN	
05	DVISIN3	0000	0000	DGTSTIN	
06	DVISIN1	0000	0000	DBTSTIN	level 1 BTM
07	DVISIN1	0000	0000	DBTSTIN	
08	DVISIN2	0000	0000	DBTSTIN	
09	DVISIN2	0000	0000	DBTSTIN	
10	DVISIN3	0000	0000	DBTSTIN	
11	DVISIN3	0000	0000	DBTSTIN	
12	DVISIN1	0000	0000	DPAININ	level 1 PM
13	DVISIN1	0000	0000	DPAININ	
14	DVISIN2	0000	0000	DPAININ	
15	DVISIN2	0000	0000	DPAININ	
16	DVISIN3	0000	0000	DPAININ	
17	DVISIN3	0000	0000	DPAININ	
18	DVISIN1	DVISIN2	0000	DGTSTIN	level 2 GTM
19	DVISIN1	DVISIN2	0000	DGTSTIN	
20	DVISIN1	DVISIN3	0000	DGTSTIN	
21	DVISIN1	DVISIN3	0000	DGTSTIN	
22	DVISIN2	DVISIN3	0000	DGTSTIN	
23	DVISIN2	DVISIN3	0000	DGTSTIN	
24	DVISIN1	DVISIN2	0000	DBTSTIN	level 2 BTM
25	DVISIN1	DVISIN2	0000	DBTSTIN	
26	DVISIN1	DVISIN3	0000	DBTSTIN	
27	DVISIN1	DVISIN3	0000	DBTSTIN	
28	DVISIN2	DVISIN3	0000	DBTSTIN	
29	DVISIN2	DVISIN3	0000	DBTSTIN	
30	DVISIN1	DVISIN2	0000	DPAININ	level 2 PM
31	DVISIN1	DVISIN2	0000	DPAININ	
32	DVISIN1	DVISIN3	0000	DPAININ	
33	DVISIN1	DVISIN3	0000	DPAININ	
34	DVISIN2	DVISIN3	0000	DPAININ	
35	DVISIN2	DVISIN3	0000	DPAININ	
36	DVISIN1	DVISIN2	DVISIN3	DGTSTIN	level 3 GTM
37	DVISIN1	DVISIN2	DVISIN3	DGTSTIN	
38	DVISIN1	DVISIN2	DVISIN3	DBTSTIN	level 3 BTM
39	DVISIN1	DVISIN2	DVISIN3	DBTSTIN	
40	DVISIN1	DVISIN2	DVISIN3	DPAININ	level 3 PM
41	DVISIN1	DVISIN2	DVISIN3	DPAININ	

The function of, the inputs to and the outputs from the 'T*-element part I' card, the 'T*-element part II' card and the 'fraction multiplier' card have been explained in Section 6.3.

As has been mentioned in Section 6.3, the correlation values are stored in the seven 'counter' cards, each having six 4-bit binary counters. The input to these counters are enabled by forty-two control signals $\overline{LD00}$ through $\overline{LD41}$ coming from the 'decoder'.

During a read-out from the MEMORY, as the control sequences through the forty-two T*-elements, everytime the stored sensory input values match the presented sensory input values, the LREAD signal goes from a "0" to a "1". This signifies that the corresponding T*-elements should output its stored correlation value. The status of the LREAD signal for the forty-two T*-elements during a read-out, is stored in a sequence in a 42-bit shift register (actually a 48-bit shift register of which the first forty-two bit positions are used) in the 'shift register' card. These forty-two bits are available as forty-two control signals, LROO through LR41, to enable the outputs of the proper counters in the 'counter' cards.

The forty-two 4-bit numbers from the seven 'counter' cards are compared in M-elements in seven 'comparison' cards to generate the nine level outputs (three from each of GTM, BTM and PM) in a manner as discussed in Section 5.1. The circuit of a 'comparison' card is functionally equivalent to the following arrangement shown in Figure 6.12. The input-output relationship of this arrangement, shown in terms of the basic continuous logic elements is fairly self-explanatory.

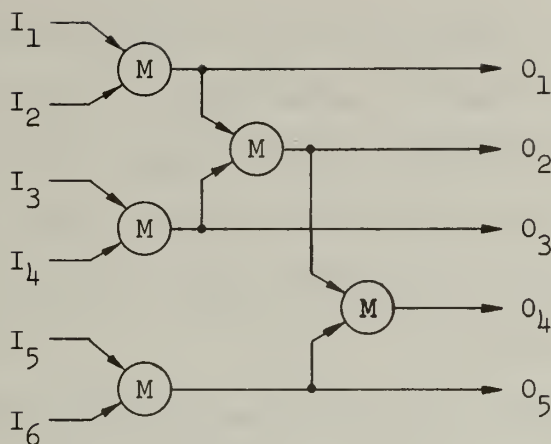


Figure 6.12 Comparison Card: Continuous Element Logic Diagram

The level outputs go to the three 'MIS' cards which realizes the MIU as detailed in Section 5.2. The six 4-bit numbers (MISOUT1 - 3, ALARMIS1 - 3) and the DNODECF signals coming out of these 'MIS' cards go into the 'max elements and DU' card. In this circuit, first the DMEMOUT and the DALARMOUT signals are generated by use of M-elements (see Figure 5.3). These signals along with DINNATESTIM, DHUNGER, DNODECF are processed in the circuit realization of the decision unit (DU) to result in the decisions (DDIN signal). The signal LPOSEQLOC entering the 'max elements and DU' circuit is a signal that indicates if the position of FROG is the same as the location of the stimulus. The DDIN signal is 4-bit wide, one bit goes to a "1" to indicate the decision in accordance with the Figure 5.5.

The circuits in the 'comparison', the 'MIS' and the 'max elements and DU' cards being purely combinational, are fairly straightforward.

The circuit cards described up to now occupy the lower rack of the FROG assembly (see Figure 3.1). The following paragraphs give description of the circuit cards in the upper rack, and other variously mounted circuits.

The signals from the switches on the front panel come to the circuit board called 'switches and RNG2'. This circuit board has a pseudo-random

number generator to generate, if required to do so by selection of the panel switches, a random sequence of stimuli at random locations. The type of the stimulus chosen for a cycle is outputted as a 3-bit signal DSTIMIN from this card. The location choice is indicated by a 2-bit number DLOCIN. The motion override signal, LOVERRIDE, and the enter pulse, PENTER, from the panel are also available as outputs of this card.

The 'register' card essentially consists of several latches for storing various signals. The HUNGER register is in the form of a 4-bit counter, which stores the current value of the hunger signal. Incrementation or decrementation of this value is done by making the counter either count up or count down. The counter is protected by additional logic from overflowing or underflowing (i.e., if it is already at 1111, an attempt to increment will not affect the count, and if a 0000, attempt to decrement will leave the value unchanged). There is a provision, a 4-bit preset switch, by which the hunger value can be initialized to a desired value. The 'register' card has the counter COUNTER2 which is used for keeping count when sequencing through the forty-two T*-elements (refer to Section 6.3). The outputs of this counter are the six bits of DTSTARNO. The STIMULUS register loads, at the beginning of a trial, the value of the DSTIMNO signal. This allows the switches to be changed for the next trial, while computation for the current trial is in progress.

The decision, as generated in the decision unit, comes to the 'register' card as the DDIN signal bits and is stored in two separate registers. The two registers, the DECISION register and the DECISION DISPLAY register, are both loaded at the end of the read cycle, when the DDIN signal becomes reliable. The reason for having two registers holding the same quantity is that when a wrong decision (i.e., a decision to feed on a dangerous stimulus) is made, then following the encounter with the stimulus the DECISION DISPLAY

register is changed to show a flee, while the initial wrong decision is maintained in the DECISION register for reference.

The output from the DECISION DISPLAY register and that from the STIMULUS register decoded to six lines (with one active at a time) are sent via cables to the 'monitor' circuit for the display of the decision (1 of 4) and the stimulus number (1 of 6). The 'monitor' card is mounted behind the panel.

The 'lighting and motion I' card has the circuitry which decides which lamp segments to light up on the board based on i) the position of FROG, ii) the location of the stimulus and iii) the decision about the stimulus. The current position of FROG is maintained in a 2-bit POSITION register in this circuit card, and updated at appropriate times. The location of the stimulus, as indicated by the DLOCIN signal from the 'switches and RNG2' card, is stored for a trial in another 2-bit register. This card also generates a command signal, PSTARTMOVE, for initiating the movement of the movable block (MB) at appropriate times. Lastly, the card has a 3-digit BCD counter called COUNTER1 which keeps track of the trial number.

From the 'lighting and motion control I' card, therefore, the pulse command signal PSTARTMOVE goes to the 'motion control II' circuit. The DPOSITION (2-bits) and DLOCATION (2-bits) signals out of the appropriate registers, and the DTRIALNO signal (three 4-bit groups) go on cables to the 'monitor' card for display of the position of FROG, the location of the stimulus and the trial number. Also, two more 2-bit signals, LSTART and LFINISH, generated in this card, go on a different cable to the 'lighting control II' circuit to indicate the starting and the end position of FROG every time a motion has to be initiated.

The 'motion control II' circuit controls the movement of MB. The PSTARTMOVE command sets a flipflop in the circuit. The output LMOVE of the flipflop is delayed by one second to allow for the lamps on the board, when

lit up or changed, to settle to a stable pattern. This delayed signal LM goes via a flexible cable to the circuits in MB to start the motors. The stop signal $\overline{LP4}$ (active low), which is generated as soon as the end of the track is detected (see Section 3.7), is relayed back on this cable to reset the flipflop. This immediately deactivates LMOVE and LM signals to stop MB. The dc power to the two motors are also provided from this card to MB through the cable. Two potentiometers can be adjusted for partial control of the currents in the individual motors. This helps in balancing the speeds of the two motors.

The 'lighting control II' card is a 17 inch wide card which is mounted below the lower card rack (see Figure 3.1). It has twenty six circuits for driving lamps. Eighteen of these are for driving segments of under-the-track lights, four for stop lights indicating end of track, and four lights for indicating the four locations on the board. The signals received by 'lighting control II' circuit through the cable from 'lighting and motion control I' are DLOCATION (2-bits), DPOSITION (2-bits), DSTART (2-bits) and DFINISH (2-bits). These signals are decoded to light up the appropriate lights and light segments.

The 'monitor' card is mounted at the back of the panel. The DTRIALNO signal (three BCD digits) is displayed on three BCD 7-segment LED displays as the TRIAL NUMBER. The hunger value is displayed on a linear display with fifteen LEDs. The DPOSITION signal is displayed on a 7-segment LED display to indicate the current position of FROG. The DLOCATION signal is similarly displayed on another 7-segment LED display. The stimulus number, given by the 6-line $\overline{DSTIMULUS}$ signal, and the decision, given by the set of four signals ($\overline{DREST}$, $\overline{DALARM}$, $\overline{DFLEE}$, $\overline{DFEED}$) come to the 'monitor' card

from the 'register' card. These are displayed by LED lamps. The circuit in the 'monitor' card is purely combinational.

The remaining four cards, 'system clock and busy flip-flops', 'control I', 'control II', and 'control III', generates the necessary control signals for the proper sequencing of all operations.

7. RESULTS AND CONCLUSION

7.1 Discussion of a Typical Sequence of Trials

In order to investigate the behavior of FROG, the machine was run through a sequence of 50 trials, in which all the stimuli were presented in a fairly arbitrary fashion. A trial by trial summary of the run is given in Appendix A. This section gives a discussion of the results.

The run started with three consecutive presentations of stimuli of type 5, which are dangerous for FROG. In the very first presentation (trial #1), FROG ignored this stimulus because it knew nothing about the stimulus, and it was not hungry enough ($\text{hunger} = 7/15 < 1/2$). In the next trial, since hunger value was greater than 0.5 ($\text{hunger} = 8/15$), FROG was tempted to feed on the stimulus. The encounter was obviously painful, and so, FROG had to flee from the stimulus. The single encounter provided some learning in level 3, which made FROG correctly recognize a stimulus of type 5 to be a dangerous one in trial #3. In spite of the fact that the hunger value was high, FROG avoided the stimulus in this trial.

For the next three trials, FROG was presented with stimuli of type 1, which are good tasting. The presented stimulus was fed upon each time. The first time (trial #4), the decision was hunger-forced. But the next two times, the decisions were memory assisted, with the cognitive mechanism making correct inferences.

Following this, a stimulus of type 5 was presented. FROG decided to feed on it. This obviously was a wrong decision. This happened because of the shared feature (the color, black) between a type 1 stimulus and a type 5 stimulus. The encounters with type 1 stimulus in trial #s 4, 5 and 6 had biased it into generally associating the color black with good taste. So, in

trial #5, on seeing a black stimulus FROG concluded that it must be good tasting (a level 1 decision) without paying much attention to the particular combination in which this feature appears. A previous encounter with a stimulus of type 5 (trial #2) was not of help, because it did not provide sufficient counteracting evidence in level 1 (the most general level). This trend continued in the next trial (#8), when a wrong attempt to feed on a type 5 stimulus was once again made in confusion with a type 1 stimulus. This trend got corrected in the next trial. The stimulus was correctly recognized and avoided.

Presentation of a stimulus of type 1 in the 10th trial was followed by a feeding action. Since it was the fourth time a stimulus of this type was eaten, even the level 1 T*-element outputs were brought up to their full potential. So a full learning of stimuli of type 1 was achieved at the end of this trial.

A type 3 stimulus was presented for the first time in trial #11. It is red, round and small. FROG did not have an encounter with anything red and/or small before, but did have some unpleasant experience with something round (stimulus of type 5). So FROG's reaction to a type 3 stimulus was one of avoidance. Obviously no learning took place.

A stimulus of type 4 was introduced for the first time in trial #12. Except for its color, which is yellow, it looks very similar to a stimulus of type 1. A type 4 stimulus is a bad tasting stimulus. Because of the similarity of stimuli of types 1 and 4 and because of the pleasantly rewarding experience with type 1 stimuli, FROG decided that the new stimulus must be good tasting too and fed on it. The same behavior was witnessed in the next two trials as well. Because of these feedings, the situation was corrected in trial #15, when a type 4 stimulus was ignored, being recognized as a bad tasting stimulus.

Trial #16 saw the first introduction of a type 2 stimulus. Because of its yellow color, it was initially thought to be of type 4, which FROG had just learned to be a bad tasting stimulus. So the presented stimulus was ignored by FROG.

FROG was again tested with some type 5 stimuli in trial #s 17 and 18. In trial #17, FROG attempted (wrongly) to feed on it, in confusion with a stimulus of type 1, which has the same color. Although, FROG had apparently learned to distinguish a stimulus of type 5 from a stimulus of type 1 (witness trial #9), the encounter with a stimulus of type 1 in trial #10 led FROG to believe, once again, that the color black is associated with good taste. FROG's attempt to feed on a type 5 stimulus in trial #17 resulted in full learning of a type 5 stimulus.

The second presentation of a type 2 stimulus was in trial #19, when FROG recognized it to be a dangerous one. This is because a type 2 stimulus is of the same size as a type 5 stimulus, and a large number of unpleasant encounters with type 5 stimuli had already taken place by that time. But in spite of this warning from the memory, FROG decided to go ahead and eat the stimulus because the hunger condition was at its maximum (starvation point). In the next eight trials (#20 through 27), FROG was presented with type 2 stimuli only. During the trials 20 through 25, FROG consistently recognized type 2 stimuli to be dangerous ones (being biased by #5), but fed on it in trials 21, 23 and 25, being starvation forced. These four feeding attempts caused full learning of stimuli of type 2, and helped reverse the trend. This is evidenced by FROG's decision to feed on stimuli of type 2 in trials 26, 27, and later in trials 47, 48 as well.

A type 6 stimulus appeared for the first time in trial #28. Based solely on color, a type 6 stimulus is similar to a type 3 (a bad tasting) stimulus. Based on size, a type 6 stimulus is similar to both a type 2 (good tasting)

stimulus and a type 5 (dangerous) stimulus. Because of these shared features, more than one submemory (GTM, BTM and PM) outputted equal values resulting in no generated response at these levels. Since nothing was known about a stimulus of type 6, no response came from level 3 either. So, the MIU generated no output and NODECF signal was 1.0. Under this condition, FROG decided to feed on the stimulus based solely on the hunger value (hunger = 1100). This feeding attempt was obviously followed by a fleeing reaction. This encounter produced enough learning for FROG to stay away from a stimulus of type 6 in the very next trial (#29). This decision, of course, was primarily guided by the level 3 output from PM, with no response generated in levels 1 and 2 as in the last trial.

In the next eight trials (#30 through 37), stimuli of type 3 were presented seven times. In each of the first six presentations (trial #s 30, 31, 32, 33, 35 and 36), the reaction of FROG was similar to what it had been towards a type 3 stimulus in trial #11, i.e., one of recognizing it to be a dangerous one (confusing it with a type 5 stimulus, which has the same shape). Starvation forced attempts to feed, however, were made in trials 30, 32 and 36. These three feedings caused FROG to recognize a stimulus of type 3 correctly in trials 37 and 39. Although the stimulus avoided in trial #37, it was fed upon in trial #39, because the hunger value was 1.0. This feeding resulted in full learning of type 3 stimuli.

In the middle of this long spell with type 3 stimuli, a stimulus of type 4 was slipped-in in trial #34. It was correctly recognized as a bad tasting stimulus. A starvation condition, however, caused forced feeding, also resulting in a full learning of stimuli of type 4.

FROG's reactions to stimuli of type 6 were tested in trial #s 38, 40, 41, and 42. In trial #38, a type 6 stimulus was correctly recognized by

FROG as a dangerous stimulus, but, once again, a starvation condition existing caused FROG to attempt to feed on the stimulus anyway. The next time a type 6 stimulus was presented (trial #40), FROG decided to ignore it thinking it to be of the kind of a type 3 stimulus (both being red). The reaction was the same in the next trial (#41), although hunger = 1.0 forced FROG to make a feeding attempt with unpleasant outcome. The same behavior was repeated the next time as well. These attempts to feed on a type 6 stimulus, in sheer desperation, resulted in a full learning of stimuli of type 6.

The remaining trials (# 43 through 50) are significant. By trial # 42 each of the six different types of stimuli had been fed upon (or attempts had been made to do so) at least four times. All the stimuli were fully learned. In other words, the T*-elements were not going to be incremented or adjusted any further. Therefore, it is important for FROG to have recognized all the different stimuli correctly. An examination of the responses will show that this is definitely the case.

7.2 Results in Summary and Concluding Remarks

At birth the tendencies of the device are guided by the hunger value and/or the innate sensory input value of the presented stimulus. The reactions of FROG toward the various types of stimuli depend, among other factors, on the sequence in which the stimuli are encountered and the extent to which features are shared. For example, in the given run, whereas the good tasting stimuli of type 1 were fully learned in four appearances, the stimuli of type 2, which are good tasting too, had to be presented eight times before they were fully learned.

In general the good tasting stimuli have a better chance of being adapted faster, because they are more readily fed upon. Avoidance of the bad tasting or dangerous ones slows down the learning of those types.

Given a sufficient number of presentations (usually 4 to 10 appearances) each type of stimulus is learned correctly. However, because of the shared features between the different types and because of FROG's inherent tendency to make gross decisions, errors result in the learning phase.

There are three distinct types of errors made by FROG. The first is to decide to feed on a stimulus of the bad tasting or dangerous type. Errors of this type are easily corrected, for the feeding encounters result in additional learning which alleviates the problem.

The second type of error occurs when a good tasting stimulus is avoided. This response is not self-corrective; with feeding inhibited, additional learning is not possible. Only starvation forced feeding helped reverse the situation in this case.

A third type of error is made when a dangerous stimulus is avoided, being mistaken as a bad tasting stimulus or vice-versa. Although this behavior inhibits proper learning, its effect is not altogether undesirable. However, since this type of error means avoidance, the stimulus types are not learned properly and might subsequently lead to type 1 error.

Admittedly the device is not always perfect; it makes mistakes. It is these mistakes which make the simulation more realistic, showing that "processing from the general to the specific" is a very plausible mode of operation of the nervous systems in the biological world.

In this simulation, for the particular choice of the stimulus types, an examination of trials 43 through 50 show that except for stimulus of type 2 and 4, which were resolved only by examining the level 3 outputs, all others were resolved at level 2. If the mechanism makes a sequential search through the data in its world model (i.e., memory) during a read cycle (as would be the case when

the mechanism has only one processing element), we can argue that there will be an improvement in the response time, when attempts are always made to make decisions on a gross context. The response time is the time between the presentation of the stimulus and the making of a decision about it. It is not possible to exactly calculate the improvement in response time by organizing data in various extra levels (of generalization), because the improvement is data dependent. But, in most cases, where some general trends and natural clusters exist in the stimulus range, an improvement of the response time is to be expected. This point is further elaborated in Appendix B.

Note must be taken of the fact that generalization from the observed data gives the ability to respond to new stimuli. In this simulation, similar stimuli, however, were purposely made to be of absolutely different nature. Therefore, memory assisted approach to a new stimulus usually meant incorrect prediction (cf., trial #s 11, 12, 16). This facility can, of course be conceivably exploited more for the good of the mechanism in a different situation.

In order to respond to new situations, the scheme uses extra memory over a scheme in which the observed data is retained in a simple ordered list. Although the extra memory requirement is not of an alarming proportion (see Appendix B), it is probably possible to reduce the requirement by employing a few simple tricks. One method that seems to be of promise, is to release the memory elements (and the related data) which are not used during read-outs over an extended period of time. Thus, for those stimuli which are recognized by considering the level 1 or level 2 outputs, data relevant to such stimuli in the lower levels may be discarded. Alternatively, all memory elements may, periodically (e.g., every trial), have their contents reduced by a small amount, unless they are read out or written into in that period. Also, the elements may then be released if their contents become too small. This

can well simulate the decay or loss of memory observed in living beings. No attempt has been made to incorporate this in FROG.

The operation of FROG is sensitive to the assignment of values to the features. A proper discrimination of the various classes of stimuli is dependent on the adjustment of these values. So, the learning process in FROG is a supervised learning. Although it is possible to propose methods of statistically adapting to the proper values, schemes using unsupervised learning have generally been found to have unsatisfactory results. Incorporation of built-in premises and biases, or taking a heuristic approach, seems to provide better results in most machine cognition activity. It might be added that even in living organisms, a lot of knowledge and innate tendencies are handed down from generation to generation. This may be thought to belong to the group intelligence of the species, which an individual specimen of the species does not totally adaptively acquire from its own experience.

Other situations, where this type of modelling (using generalizations and using quick decision making based on the principle of "processing from the general to the specific") will probably be quite suited, are as follows:

(a) A complex system which is required to respond quickly to some critical modes of failure, where an exhaustive list of failure patterns is either too long to search, or cannot be completely specified. Such systems probably have to take some quick precautionary measures based on inexact judgements, and then go into further details to resolve the cause of failures. Since all modes of failure cannot be predicted and simulated ahead of time, an adaptive system becomes necessary. This might find use in automatic maintenance of a complex network or system, such as a large electronic switching systems in telephone networks.

(b) An interactive advisory system where machines are employed to advise and guide human beings in locating problems based on a pattern of symptoms. It is conceivable that the machine, at all stages, will probably need to make inexact decisions about probable causes, and ask the human subject meaningful questions to get to the final interpretation. A medical diagnostic program can probably be thought in these lines.

APPENDIX A

SUMMARY OF A SEQUENCE OF TRIALS

The sequence consisted of 50 trials. A summary of the run is given in the following pages. A guide to the various columns is given below:

Trial# - serial number of the trial.

Stimulus - the number of the stimulus presented in the current trial.

HUNGER - current value of FROG's hunger.

FEED - value of the FEED signal.

ALARMOUT - value of the ALARMOUT signal.

Decision - mode of action decided upon by FROG for the stimulus presented. Abbreviations used are:

F = FROG decided to feed on the stimulus.

R = FROG decided to rest and ignore the stimulus.

A = FROG is alarmed by the presence of the stimulus.

FF = FROG decided to feed on a dangerous stimulus, and
flee to its safe resting place on encounter.

Primary Reason - Entries to this column are the signals which were primarily responsible for the decision. The following abbreviations are used:

h = HUNGER.

i = INNATESTIM.

GTM_r = level r GTM output.

BTM_r = level r BTM output.

PM_r = level r PM output.

<u>Trial#</u>	<u>Stimulus</u>	<u>HUNGER</u>	<u>FEED</u>	<u>ALARMOUT</u>	<u>Decision</u>	<u>Primary Peason</u>	<u>Comments</u>
1	5	0111	0111	0000	R	h<1000	
2	5	1000	1000	0000	F	h=1000	
3	5	1010	0000	1110	A	PM3=1110	trend corrected
4	1	1011	0000	0000	F	h=1011	
5	1	1010	1010	0000	F	GTM3=1101, h=1010	
6	1	1001	1001	0000	F	GTM2=1010, h=1001	
7	5	1000	1000	0000	FF	GTM1=1010, h=1000	confusion with a 1
8	5	1010	1010	0000	FF	GTM1=1010	confusion with a 1
9	5	1100	0000	1110	A	PM2=1110	trend corrected
10	1	1101	1010	0000	F	GTM1=1010	full learning of 1's
11	3	1100	0000	1000	A	PM1=1000	confusion with a 5
12	4	1101	1100	0000	F	GTM1=1100	confusion with a 1
13	4	1100	1100	0000	F	GTM1=1100	confusion with a 1
14	4	1011	1011	0000	F	GTM1=1100, h=1011	confusion with a 1
15	4	1010	0011	0000	R	BTM3=1100	trend corrected
16	2	1011	0011	0000	R	BTM1=1100	confusion with a 4
17	5	1100	1100	0000	FF	GTM1=1101, h=1100	confusion with a 1, full learning of 5's

<u>Trial#</u>	<u>Stimulus</u>	<u>HUNGER</u>	<u>FEED</u>	<u>ALARMOUT</u>	<u>Decision</u>	<u>Primary Reason</u>	<u>Comments</u>
18	5	1110	0000	1110	A	PM1=1110	trend corrected
19	2	1111	1110	1110	F	PM1=1110,i=1110,h=1111	confusion with a 5, starvation forced feeding
20	2	1110	0000	1110	A	PM1=1110	confusion with a 5
21	2	1111	1110	1110	F	PM1=1110,i=1110,h=1111	confusion with a 5, starvation forced feeding
22	2	1110	0000	1110	A	PM1=1110	confusion with a 5
23	2	1111	1110	1110	F	PM1=1110,i=1110,h=1111	confusion with a 5, starvation forced feeding
24	2	1110	0000	1110	A	PM1=1110	confusion with a 5
25	2	1111	1110	1110	F	PM1=1110,i=1110,h=1111	confusion with a 5, starvation forced feeding, full learning of 2's
26	2	1110	1110	0000	F	GTM2=1110	trend corrected
27	2	1101	1110	0000	F	GTM2=1110	
28	6	1100	1100	0000	FF	h=1100	confusion with 2,3,5
29	6	1110	0000	1111	A	PM3=1111	trend corrected
30	3	1111	1111	1011	F	PM1=1011,h=1111	confusion with a 5, starvation forced feeding
31	3	1110	0000	1011	A	PM1=1011	confusion with a 5

<u>Trial#</u>	<u>Stimulus</u>	<u>HUNGER</u>	<u>FEED</u>	<u>ALARMOUT</u>	<u>Decision</u>	<u>Primary Reason</u>	<u>Comments</u>
32	3	1111	1111	1011	F	PM1=1011,h=1111	confusion with a 5, starvation forced feeding
33	3	1110	0000	1011	A	PM1=1011	confusion with a 5
34	4	1111	1100	0000	F	BTM3=1100,i=1100,h=1111	starvation forced feeding, full learning of 4's
35	3	1110	0000	1011	A	PM1=1011	confusion with a 5
36	3	1111	1111	1011	F	PM1=1011,h=1111	confusion with a 5, starvation forced feeding
37	3	1110	0000	0000	R	BTM2=1111	trend corrected
38	6	1111	1111	1111	FF	PM3=1111,h=1111	starvation forced feeding
39	3	1111	1111	0000	F	BTM2=1111,h=1111	starvation forced feeding, full learning of 3's
40	6	1110	0000	0000	R	BTM1=1111	confusion with a 3
41	6	1111	1111	0000	FF	BTM1=1111,h=1111	confusion with a 3, starvation forced feeding
42	6	1111	1111	0000	FF	BTM1=1111,h=1111	confusion with a 3, starvation forced feeding, full learning of 6's
43	1	1111	1101	0000	F	GTW2=1101	
44	1	1110	1101	0000	F	GTW2=1101	
45	3	1101	0000	0000	R	BTM2=1111	

<u>Trial#</u>	<u>Stimulus</u>	<u>HUNGER</u>	<u>FEED</u>	<u>ALARMOUT</u>	<u>Decision</u>	<u>Primary Reason</u>	<u>Comments</u>
46	4	1110	0011	0000	R	BTM3=1100	
47	2	1111	1110	0000	F	GTM2=1110	
48	2	1110	1110	0000	F	GTM2=1110	
49	5	1101	0000	1110	A	PM2=1110	
50	6	1110	0000	1111	A	PM2=1111	

APPENDIX B

MEMORY REQUIREMENT AND RESPONSE TIME FOR THE PROPOSED SCHEME

Consider the memory of a cognitive scheme as proposed in Chapter 2, with m sensory inputs and one reward parameter. Let the sensory inputs and the reward parameter take on n possible non-zero values in the interval 0 to 1. Then,

$$\begin{aligned}
 N_r &= \text{the number of memory (T*) elements in level } r. \\
 &= \text{the number of different combinations of } m \text{ sensory inputs taken } r \\
 &\quad \text{at a time } x \text{ (number of possible values for each sensory input)}^r \\
 &\quad x \text{ number of possible values of the reward input.} \\
 &= \binom{m}{r} x n^r \times n = \binom{m}{r} \cdot n^{r+1}.
 \end{aligned}$$

$$\text{Total memory requirement} = N_1 + N_2 + \dots + N_r + \dots + N_m = \sum_{r=1}^m \binom{m}{r} \cdot n^{r+1}.$$

In the special case of $m = 3$ and $n = 15$ (as in FROG), we have three levels, and

$$N_1 = \text{number of elements required in level 1} = 3 \times 15^2 = 675,$$

$$N_2 = \text{number of elements required in level 2} = 3 \times 15^3 = 9825,$$

$$N_3 = \text{number of elements required in level 3} = 15^4 = 50625.$$

The complete knowledge about all stimuli is maintained in a discrete manner in the level 3 elements. The data in the other levels are derived by induction from the data in level 3. So extra memory requirement

$$\begin{aligned}
 &= \frac{N_1 + N_2}{N_3} \times 100\% \\
 &= \frac{675 + 9825}{50625} \times 100\% = 21.4\%
 \end{aligned}$$

If the recognition scheme works only on the entries of level 3 (which contains the complete experience), the response time, R_{old} , may be thought

to be proportional to the length of the list. Thus,

$$R_{\text{old}} = 50625 \text{ time units.}$$

The response time in the new scheme, R_{new} , with three levels depends, of course, on the level at which a decision is made.

$$R_{\text{new}} = \sum_{x=1}^r N_x \text{ time units, if resolved in level } r.$$

$$= 675 \text{ time units, if } r = 1.$$

$$= 10500 \text{ time units, if } r = 2.$$

$$= 61125 \text{ time units, if } r = 3.$$

The best situation exists when all stimuli can be resolved in level 1.

Then the reduction in response time is $(50625 - 675) \cdot 100 / 50625 = 9.5\%$!

The worst situation exists when it is always necessary to go to level 3 elements for a decision. In this case, the response time is increased by

$$(61125 - 50625) \cdot 100 / 50625 = 20.6\%.$$

REFERENCES

- [1] Fink, D. G., Many Faces of Intelligence, The World of Computers, John Diebold: Editor, Random House, New York, pp. 415-419, 1973.
- [2] Preparata, F. P., Foundations and Methodologies of Image Recognition. Pictorial Artificial Intelligence, unpublished collection of class notes, University of Illinois, Chapter 0, 1972.
- [3] Minsky, M., Papert, S., Artificial Intelligence - Progress Report, A. I. Lab. Memo. No. 252, Massachusetts Institute of Technology, January 1972, Section 5.1.1.
- [4] Piaget, J., The Psychology of Intelligence, Rutledge and Paul, London, 1950.
- [5] Reitman, W. R., Cognition and Thought, John Wiley and Sons, New York, 1965.
- [6] Lenneberg, E., Biological Foundations of Language, John Wiley and Sons, New York, 1967.
- [7] Oettinger, A., Language and Information, American Documentaiton, vol. 19, no. 3, July 1968, p. 298.
- [8] Zadeh, L. A., Fuzzy Sets, Information and Control, vol. 8, pp. 338-353, 1965.
- [9] Maturana, H. R., Lettvin, J. Y., McCulloch, W. S., Pitts, W. H., Anatomy and Physiology of Vision in the Frog (Rana pipiens), J. G. Physiology, 43, pp. 129-175, 1960.
- [10] Wo, Y. K., APE Machine: A Novel Stochastic Computer Based on a Set of Autonomous Processing Elements, Department of Computer Science Report No. 556, University of Illinois, Section 3.2 and 4.5, pp. 15-16 and 49-50, February 1973.

VITA

Debasish Bose was born in Calcutta, India, on January 30, 1949. He received his B. Tech. (Hons.) in Electronics and Electrical Communication Engineering in June 1970 from the Indian Institute of Technology, Kharagpur, India. He came to the University of Illinois in September 1970 as a Graduate Teaching Assistant in the Department of Electrical Engineering. Since June 1972, he has been employed by the Department of Computer Science both as a Research and as a Teaching Assistant. He joined the Information Engineering Laboratory of the Department of Computer Science as a research member in March 1971 to work under Professors S. R. Ray and W. J. Poppelbaum. He received his M.S. in Electrical Engineering in June 1973.

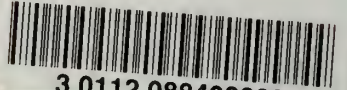
He is a member of the Computer Group of IEEE.

BIBLIOGRAPHIC DATA SHEET		1. Report No. UIUCDCS-R-76-824	2.	3. Recipient's Accession No.	
Title and Subtitle FROG (FORMULATION OF A ROBOT ORGANIZATION THAT GENERALIZES): A HARDWARE EXPERIMENT IN MACHINE COGNITION				5. Report Date August 1976	
Author(s) Debasish Bose				6.	
Performing Organization Name and Address Department of Computer Science University of Illinois at Urbana-Champaign Urbana, Illinois 61801				8. Performing Organization Repr. No. UIUCDCS-R-76-824	
Sponsoring Organization Name and Address Department of Computer Science University of Illinois at Urbana-Champaign Urbana, Illinois 61801				10. Project/Task/Work Unit No.	
				11. Contract/Grant No.	
Supplementary Notes				13. Type of Report & Period Covered Ph.D. Thesis	
				14.	
Abstracts This paper develops a scheme for machine representation of knowledge for artificial cognition of stimuli characterized by multi-parameter feature (sensory input) vector, with a view to achieving (i) adaptive behavior (ii) ability to handle new situations and (iii) fast response. The scheme proposed includes a learning strategy which organizes the experience with known inputs in various levels of inductive generalization, and a recognition strategy which makes the processing of stimulus for cognition proceed from the most general context towards a more detailed and specific search. The limitation, of course, is the occasional occurrences of mistakes due to over-generalization. The sensory inputs are allowed to assume values in the signal continuum 0 to 1. The formulation uses the concept of continuous of multivalued logic. The scheme is exemplified by simulating the feeding habits of a small animal in an actual hardware demonstration project called FROG (Formulation of a Robot Organization that Generalizes) FROG is a mini-robot which sustains existence by recognizing sources of potential food supply and hazards. The adaptive development of the cognitive memory of the device shows interesting self-preservative behavior.					
7. Key Words and Document Analysis. 17a. Descriptors					
Artificial Intelligence		reward			
Cognitive Mechanisms		response			
world model		decisions			
stimulus		recognition cycle			
sensory inputs		learning cycle			
b. Identifiers/Open-Ended Terms					
c. COSATI Field/Group					
Availability Statement Release Unlimited		19. Security Class (This Report) UNCLASSIFIED		21. No. of Pages 102	
		20. Security Class (This Page) UNCLASSIFIED		22. Price --	

DEC 15 1976



UNIVERSITY OF ILLINOIS-URBANA
510.84 IL6R no. C002 no.824-829(1976
Internet report /



3 0112 088402992